DISSERTATIONES
INFORMATICAE
UNIVERSITATIS
TARTUENSIS
34

KRISTJAN KRIPS

Privacy and Coercion-Resistance in Voting

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

34

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS
34

KRISTJAN KRIPS

Privacy and Coercion-Resistance in Voting

UNIVERSITY OF TARTU
Press
1632

Institute of Computer Science, Faculty of Science and Technology, University of Tartu, Estonia.

Dissertation has been accepted for the commencement of the degree of Doctor of Philosophy (PhD) in Computer Science on April 8, 2022 by the Council of the Institute of Computer Science, University of Tartu.

Supervisors

Dr. Jan Willemson
Cybernetica AS
Tartu, Estonia

Dr. Sven Laur
University of Tartu
Tartu, Estonia

Opponents

Prof. Dr. Olivier Pereira
Université catholique de Louvain
Louvain-la-Neuve, Belgium

Prof. Dr. Carsten Schürmann
IT University of Copenhagen
Copenhagen, Denmark

The public defense will take place on June 8, 2022 at 12:15 in Narva mnt 18-1021 and via Zoom.

The publication of this dissertation was financed by the Institute of Computer Science, University of Tartu.

ISSN 2613-5906
ISBN 978-9949-03-869-5 (print)
ISBN 978-9949-03-870-1 (PDF)

University of Tartu Press
`http://www.tyk.ee/`

To freedom

ABSTRACT

The introduction of the Australian ballot in the middle of the 19th century led the way for ballot secrecy becoming a requirement for conducting free and fair elections. Since that time, the advancement of technology has made it possible to use electronic voting systems to cast and tally votes. While convenient, such systems also make it more difficult to observe the elections. Thus, to assure that the electronic voting systems function correctly, they must be made transparent and auditable. However, the measures used to provide integrity and transparency are often incompatible with the requirement of ballot secrecy. This can lead to a conflict between the security requirements set for elections, which is one of the aspects that is studied in this thesis.

A significant threat to the freedom of vote is coercion, which can take the forms of bribery and intimidation. While in traditional paper-based voting systems, voter's privacy and coercion-resistance can be provided with the help of a voting booth, this mitigation measure is not available for remote voting systems. Thus, if the vote is cast via a postal voting system or a remote online voting system, it becomes more challenging to provide coercion-resistance to the voter.

We analysed seven recently proposed remote online voting systems to assess whether their coercion-resistance techniques could be applied in real-world conditions. The situation is further complicated due to many voting schemes being designed to be end-to-end verifiable, making it possible to check whether the ballots were tallied correctly and, in some cases, also whether eligible voters cast the votes. Our analysis revealed that the systems which aim to simultaneously achieve coercion-resistance and end-to-end verifiability depend on multiple non-trivial assumptions that are difficult to implement in practice.

Although much of the voting research has focused on electronic voting systems, the advancement of technology has also impacted the security of traditional paper-based voting. We show that the side-channels which commonly affect electronic systems are also relevant for paper ballots. More specifically, we introduce two proof-of-concept attacks against ballot secrecy that are based on an acoustic side-channel. However, ballot secrecy can also be violated if coercers request voters to use smartphones to record their votes. Thus, it becomes evident that traditional voting systems can not guarantee absolute ballot secrecy.

The designers of the Estonian internet voting (i-voting) system also had to find a compromise between the transparency and coercion-resistance requirements. We give an overview of some of the design choices and describe some of the existing issues in the Estonian i-voting system. In addition, we make recommendations for improving its integrity and privacy guarantees. One of these ideas manifested itself in creating the first independent microcontroller-based voting client for the Estonian i-voting system. As the final contribution, we evaluated the security impact of introducing a smartphone-based voting client for the Estonian i-voting system.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Democracy originates from ancient Greece, where citizens in some regions were able to directly vote over decisions. As direct voting does not scale well, over time, a solution was to elect representatives. However, the specific way how the representatives are elected depends on the used election system. In turn, the election systems differ from country to country as they depend on legislation and cultural background.

The requirements and ways how elections are organised have changed through time. A prominent example is voter's privacy, which has not always been a requirement. For instance, in France, ballot secrecy was introduced into the Constitution of 1795 but was not fully implemented in practice until 1913 [103]. While legislation allowing ballots to be used in elections was available in the US and France in the middle of the 19th century, it did not prevent voters from being observed during vote casting [55]. For instance, it was possible to monitor voters to check which ballot sheet they picked up. This was possible as candidates were able to print their own distinct ballot sheets. Thus, by watching voters pick up and fill uniquely designed ballots, bribers could get proof of the bribed voters holding up their end of the bargain.

In Britain, the political debate regarding the introduction of the secret ballot was initiated by Jeremy Bentham at the end of the eighteenth century [305]. However, Britain was not the first to introduce the secret ballot requirement into legislation. The election law regarding the secret ballot was revolutionised in Australia in the middle of the 19th century. In 1856, the first states of Australia adopted the new interpretation of the secret ballot [235]. Due to the specific requirements for the ballot, it became known as the Australian ballot.

The requirements for the secret ballot that we are used to in the modern world have their roots in the four main requirements of the Australian secret ballot [129]. First, the ballots were printed and paid for by the election organiser. Second, all officially nominated candidates were listed on the ballot. Third, ballots were distributed only by election officials at polling stations. Fourth, the voters were provided with an environment to fill in the ballots privately.

Other countries followed Australia in reforming the way how votes were cast. Britain introduced similar requirements for a secret ballot in 1872, followed by Canada in 1874 [103, 57]. Kentucky was first in the United States to trial the Australian ballot in 1888, but the first statewide elections with the new requirements were held in Massachusetts in the same year [129].

As holding elections with the secret ballot is less convenient for the organisers, it can be deduced that introducing a more robust version of ballot secrecy was a necessity. One of the main reasons was to prevent voters from being bribed. For example, in the second part of the 19th century, it was common in some parts of New York for voters to be accompanied by party workers to check how they voted [273]. Britain also had issues with bribery that led to the reform of the

election legislation in the second part of the 19th century [276]. These issues are illustrated by the debates in the House of Commons in 1867, which described corruption and bribery in Great Yarmouth and Lancaster during the elections in 1865 [2]. In the former, close to 32% of the voters had taken bribes, and in the latter, 64% of the voters were claimed to be corrupt.

While direct bribery is not a significant issue in some democratic societies, the threat of coercion and bribery has not disappeared [284]. Thus, it is common for voting systems designed for political elections to provide both ballot secrecy and coercion-resistance. It has to be understood that ballot secrecy and coercion-resistance can be seen as a means for guaranteeing free and fair elections, thereby making it possible for eligible voters to freely express their will and ensuring that all voters' votes carry the same weight [217]. The significance of privacy and the necessity to provide additional coercion-resistance measures also depends on the election system and cultural aspects. For example, in Switzerland, coercion is not considered a significant issue [264].

Over time, the threats to voting systems have transformed by following the evolution of technology and changes in society. It has become possible to violate ballot secrecy with recording equipment available to the vast majority of voters in the form of a smartphone. In addition, the introduction of electronic voting technologies has increased the risk of voters' privacy being violated. While the threats to electronic voting technologies have gotten much focus by the researchers, there is less research about the effects of modern technology on traditional paper-based ballots.

1.1. Elections and vote privacy

By filling in a ballot in a voting station, the voter is provided with an environment that is isolated from the prying eyes of other humans. However, this voting method is not the optimal one for all voters.

For example, the voters who live abroad or in remote locations may not be able to visit the voting stations. If no remote voting methods are provided, these voters are not able to participate in the elections. Besides the issues caused by living in a remote location, there are also other factors to consider. One of these issues is related to the privacy aspects when assisted voting or proxy voting has to be used. Therefore, disabled voters can have more privacy when voting from home via a computer compared to delegating permission to cast a paper vote or being helped to cast a vote.

Traditional voting systems do not provide the means for every voter to verify that their vote was included in the tally and that all ballots were correctly counted. However, such guarantees can be provided by voting systems that rely on cryptography to offer verifiability [188, 50].

In general, the security requirements of a voting system depend on the election-specific context and are affected by what is being decided or voted upon (see

Section 2.1 for an overview of election-specific security requirements).

Designing a voting system for political elections is a complex task due to the conflict between coercion-resistance and the transparency of the voting system. On the one hand, the voter should not be able to prove how the ballot was filled in to prevent vote-selling and coercion. On the other hand, to increase the trustworthiness of the elections, the voter needs assurance that the vote was included in the tally. In addition, the voting systems used for political elections have to be usable by regular voters [188]. Thus, the designers of a voting system have to find a way to resolve the conflict mentioned above.

In general, there are two main ways to manipulate the voter into voting for a specific candidate or abstaining from voting. First, some benefits, i.e., bribes, can be offered to the voter. Second, the voter can be intimidated and threatened with a punishment. Either way, coercive behaviour is more effective when the coercer has the means to check how the vote was cast. This does not mean that a proof for the cast vote is required for the coercion to succeed, as human psychology can play a role in favouring the coercer. For example, the voter may have difficulties lying about the cast vote when confronted by a coercer who is also a family member [109].

While coercive behaviour cannot be eliminated entirely, voting systems can be designed to reduce the risk of coercion to enable fair elections. This can be done by making it possible for voters to evade coercion. If the voting system provides measures that mitigate the threat of coercion, the voting system is said to be coercion-resistant. For example, many election systems incorporate voting protocols that are receipt-free, meaning that the voter is not provided with means to prove how the vote was cast. However, relying on the voting protocol to be receipt-free may not be sufficient as the voter or coercer can have other means for capturing the information on the filled-in ballot [195].

The academic community has proposed many i-voting schemes, but only a few have had trials in real political elections. One of the reasons lies in the difficulty of balancing the requirements of coercion-resistance, verifiability and usability. Several i-voting schemes try to find the balance by introducing exotic means to mitigate coercion. For example, we analysed voting schemes that aim to mitigate coercion by relying on fake credentials, specific hardware, a non-trivial registration process, and anonymous communication channels [195]. Our research showed that it is difficult to implement some of the coercion mitigation measures in practice, especially during political elections where every eligible voter must be able to participate.

In the case of paper-based voting, voters' privacy is protected in the polling station with the help of a polling booth that hides the process of filling in the ballot. Incidentally, the protective environment offered by the polling booth provides cover for a voter to record the vote casting process. The advancement of technology has made it possible for voters to have smartphones and other small smart devices that can be brought into the polling booth. Although recording the filled-

in ballot is not allowed in some countries, for example, in Germany[1] and Canada,[2] it is difficult to enforce this requirement. Therefore, vote privacy can be violated by taking a photo of the filled-in ballot or doing a live broadcast of the event. When such behaviour is forced, it can efficiently prevent the voter from having free will in deciding who to vote for. As a partial countermeasure, the coerced voter may invalidate the ballot or not submit the ballot to the ballot box. However, such a mitigation measure becomes less of an option if the voter is convinced that the coercer is monitoring the polling station.

In addition to direct coercion, there may be side-channels that can leak information about voters' choices. For example, a pen in the polling booth may contain sensors that reveal information about the cast votes [71]. It has been demonstrated that motion data generated by a smartwatch can be used for text recognition [31]. Moreover, our research has shown that the sound of filling in a paper ballot can reveal information about voter's choices [196, 197].

The issues of side-channels are also present in electronic voting systems. A famous example originates from the Netherlands, where the Nedap ES3B voting machines leaked the name of a party the voter voted for due to the leakage of electromagnetic emanation when the screen of the voting machine was switched to a different mode [142].

Many of the security issues of voting machines are related to the business model of the companies producing them. The voting machines' design is usually considered a trade secret, and thus, audits by third parties are not allowed. The lack of transparency makes it possible to produce voting machines that depend on unreliable software and hardware. As auditing the software running on the voting machines is not straightforward, the clients do not have many means to verify that the devices work as they are supposed to. However, once in a while, the voting machines end up in the hands of researchers who can reverse engineer them and find vulnerabilities.

There are several ways how the problem of vulnerable voting machines can be solved. One way is to create a secure hardware design and open-source it so that the vendors of the voting machines can use it. This path is taken in the SSITH program, which is funded by DARPA [270, 133, 223].

The second option to mitigate the threat of malicious voting machines is to deploy devices that produce a paper trail. The latter can be used as an input to risk-limiting audits, which can verify with a high probability whether the voting machines behaved correctly [211]. In addition, the voting machine could also print a receipt for the voter that makes it possible to check whether the cast vote was counted. For example, such an approach is used by ElectionGuard, which aims to provide end-to-end verifiability for voting machines [220]. While ElectionGuard and risk-limiting audits can be used to detect attacks against the integrity of the

[1] `https://www.gesetze-im-internet.de/bwo_1985/__56.html`

[2] `https://laws-lois.justice.gc.ca/eng/acts/e-2.01/page-33.html#s-281.6ss-(3)p-(b)ID0EBBDA`

ballots, they cannot identify attacks that target vote privacy.

The third option for resolving the issues with voting machines is not to use them at all. Instead, a remote voting system could be deployed for elections that have a low risk of coercion or where the risk of coercion can be mitigated. In that case, the main security issues lie in the insecure end-user devices that could be infected with malware. In principle, malware could cast its own vote, interfere while the voter is voting, or violate ballot secrecy by leaking how the ballot was filled. Thus, remote voting systems have to provide the voters with the opportunity to verify that their ballot reached the voting system. However, it is important to note that while verifiability creates a possibility to detect attacks, it can not prevent them. As voters' devices are used in a potentially hostile environment not controlled by the election organiser, it is impossible to guarantee that these computers are free of vulnerabilities.

One solution to this issue is to not rely on regular computers that can contain malware and instead use dedicated individual voting devices. Our experiments show that building a personal voting device is feasible given a voting system that has a public API [120].

1.2. Research questions and tasks

This dissertation focuses on the client-side of voting systems and aims to answer the following three research questions:

1. How practical are the coercion-resistance measures used in i-voting systems?
2. How has the advancement of technology impacted the ballot secrecy of traditional paper-based voting systems?
3. What are the existing risks in the client-side of the Estonian i-voting system and how can they be mitigated?

To answer these questions, we conducted theoretical literature-based studies and, in several cases, also practical experiments. We defined and solved the following tasks to answer the research questions:

1. *Study and analyse i-voting systems with respect to their coercion-resistance.* This task is solved by analysing seven i-voting schemes found in the scientific literature. We identified the assumptions and coercion-resistance measures used by these schemes. It turned out that some of these are difficult to implement in practice. The analysis is provided in Chapter 3.
2. *Investigate how acoustic side-channels could be exploited in paper-based voting systems.* This task is solved by building a proof-of-concept implementation that utilises the acoustic side-channel of filling-in paper ballots. In addition, we evaluated the accuracy of the proof-of-concept attack. The way how we solved the task is described in Chapter 4.

3. *Study what voting privacy and integrity related issues are present in the client-side of the Estonian i-voting system and analyse the potential mitigation measures.* This task was solved by analysing the technical architecture and implementation of the Estonian i-voting system. An overview of the Estonian i-voting system is given at the beginning of Chapter 5. The issues that we identified are described in Section 5.2. The possible mitigation measures are described in Section 5.3 and Section 5.4.
4. *Analyse the security impact of introducing a smartphone-based voting application for the Estonian i-voting system.* This task was solved by studying and analysing the security features of mobile browsers and operating systems (Android and iOS). As a result of the analysis, we identified multiple security risks, which could negatively impact smartphone-based voting. The results of the analysis are provided in Section 5.5.

1.3. Outline and the author's contributions

In this section, we outline the contents of the subsequent chapters and describe the author's main contributions in regard to the list of original publications used in this dissertation.

Chapter 2 gives an overview of security requirements relevant for elections and methods for protecting voting privacy. It also discusses the difficulty in finding the balance between verifiability and coercion-resistance in i-voting systems.

Chapter 3 describes the difficulties in achieving coercion-resistance in practice. The central part of the chapter focuses on analysing the coercion-resistance of seven i-voting schemes [195]. The author of the dissertation is the main author of the paper, which the analysis is based on: [195]:

- Kristjan Krips and Jan Willemson. On Practical Aspects of Coercion-Resistant Remote Voting Systems. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Bernhard Beckert, Ralf Küsters, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - 4th International Joint Conference, E-Vote-ID 2019, Bregenz, Austria, October 1-4, 2019, Proceedings*, volume 11759 of *Lecture Notes in Computer Science*, pages 216–232. Springer, 2019.

Chapter 4 focuses on side-channel attacks. While most side-channel attacks are discussed in the context of electronic devices, the problems with side-channel leakages are also present in paper-based voting systems. Namely, it is possible to listen in to what the voter is writing on the ballot sheet once the microphones are suitably placed at the voting booth. Two experiments conducted by the author demonstrated that the sound of filling-in paper ballots leaks information about

the vote. In the first experiment, we studied how to use the sound emitted while filling the ballot to recover the candidate number [196]. The description of the experiment and the findings are provided in Section 4.2. The author contributed by finding a way to identify the numbers based on the sound, collecting the audio samples, writing the scripts required for processing the data, analysing the data, and writing a significant part of the manuscript. In the second experiment, we tested whether information about the voter's choice could leak in case checkboxes are marked on a ballot [197]. The description of the experiment and the findings are provided in Section 4.3. The author's contribution was to build the setup that uses multiple microphones to capture the audio of the ballot being marked, find a way to determine the coordinates where the sound was emitted, collect test data, analyse the collected data, and write parts of the paper. The description of the aforementioned experiments and the analysis of how side-channels could be used against paper voting is based on the author's previously published work [196, 197]:

- Kristjan Krips, Jan Willemson, and Sebastian Värv. Implementing an Audio Side Channel for Paper Voting. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - Third International Joint Conference, E-Vote-ID 2018, Bregenz, Austria, October 2-5, 2018, Proceedings*, volume 11143 of *Lecture Notes in Computer Science*, pages 132–145. Springer, 2018.

- Kristjan Krips, Jan Willemson, and Sebastian Värv. Is Your Vote Overheard? A New Scalable Side-Channel Attack Against Paper Voting. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pages 621–634. IEEE, 2019.

Chapter 5 gives an overview of the Estonian i-voting system and investigates how the security of the client-side of the voting system could be improved. One possibility for improving voters' privacy is to provide alternatives to voting on a general-purpose computer that may be infected with malware. To achieve this goal, we described how a microcontroller can be used to build an unofficial voting client and how such an architecture affects the integrity and privacy of vote casting. The author contributed by kickstarting the project, supporting the development, doing the security analysis, and writing a significant part of the paper [120]. The author did not write the code for the voting client. The description of the voting client's architecture and the corresponding security analysis is provided in Section 5.4 and is based on the work published in [120]:

- Valeh Farzaliyev, Kristjan Krips, and Jan Willemson. Developing a Personal Voting Machine for the Estonian Internet Voting System. In Chih-Cheng

Hung, Jiman Hong, Alessio Bechini, and Eunjee Song, editors, *SAC '21: The 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, Republic of Korea, March 22-26, 2021*, pages 1607–1616. ACM, 2021.

Another alternative to the desktop voting application is to run the voting client on a smartphone. Compared to the microcontroller-based implementation, deploying a voting client on a smartphone would be easier and would serve a more significant fraction of the electorate. However, before implementing a voting client either as a stand-alone smartphone application or as a web application, the security challenges specific to smartphones have to be analysed. We describe these challenges in Section 5.5 and conclude that introducing a browser-based voting application is not recommended due to the numerous security issues. The author contributed most of the security analysis, testing the security features of mobile browsers, and writing most of the paper that summarised the analysis [162]. Section 5.5 is based on the author's previously published work [162]:

- Sven Heiberg, Kristjan Krips, and Jan Willemson. Mobile Voting – Still Too Risky? In Matthew Bernhard, Andrea Bracciali, Lewis Gudgeon, Thomas Haines, Ariah Klages-Mundt, Shin'ichiro Matsuo, Daniel Perez, Massimiliano Sala, and Sam Werner, editors, *Financial Cryptography and Data Security. FC 2021 International Workshops - CoDecFin, DeFi, VOTING, and WTSC, Virtual Event, March 5, 2021, Revised Selected Papers*, volume 12676 of *Lecture Notes in Computer Science*, pages 263–278. Springer, 2021.

In addition, some of the security-related shortcomings of the current Estonian i-voting system are listed in Section 5.2. Proposals for improvements are made in Section 5.3 to address the deficiencies in the Estonian i-voting system. The author contributed by analysing the Estonian i-voting system, identifying some of the shortcomings, proposing mitigation measures, and writing parts of the paper that summarised the findings [161]. Sections 5.2 and 5.3 are based on the work, which is published in [161]:

- Sven Heiberg, Kristjan Krips, and Jan Willemson. Planning the next steps for Estonian Internet voting. In Robert Krimmer, Melanie Volkamer, Bernhard Beckert, Ardita Driza Maurer, David Duenas-Cid, Stéphane Glondu, Iuliia Krivonosova, Oksana Kulyk, Ralf Küsters, Peter Roenne, Beata Martin-Rozumilowicz, Mihkel Solvak, and Oliver Spycher, editors, *Fifth International Joint Conference on Electronic Voting E-Vote-ID 2020 : 6-9 October 2020: Proceedings*, pages 82–97. TalTech Press, 2020.

2. PRELIMINARIES

The general requirements for conducting elections tend to be the same when considering democratic elections. While it is common for election systems to protect ballot secrecy and ensure that the ballots are correctly counted, the specific security measures depend on the election-specific context.

In general, voters have to be able to freely participate in elections without being coerced. Thus, third parties must not be able to check how voters voted. The basis for that is provided by ballot secrecy. Thereby voter's choice has to remain secret, and it must not be possible to link a voter's identity with the vote on the ballot. However, not all elections require ballot secrecy. For example, direct democracy is still practised in two Swiss cantons, where an open-air assembly known as *Landsgemeinde* is used to publicly cast a vote [294, 96]. Besides direct democracy, public vote casting is also commonly used by legislative bodies while deciding whether to pass laws [173]. However, when voters participate in parliamentary elections, protecting voters' privacy and thereby their freedom to vote can be as essential as the integrity requirements that are aimed to prevent tampering with the ballots and the election result.

The rest of this chapter is organised as follows. Section 2.1 gives an overview of the security requirements that are relevant for elections. Next, Section 2.2 lists the technologies and methods that are commonly used in i-voting systems to provide ballot secrecy, coercion-resistance, and integrity guarantees. It is followed by Section 2.3, which describes how encryption can be used to protect ballot secrecy and what are the accompanying risks. Section 2.4 describes the relevant threat actors. Next, Section 2.5 gives an overview of threats to voter's privacy posed by the voting environment and Section 2.6 describes different measures that can be used to guarantee the integrity of elections. Finally, Section 2.7 provides a discussion on the conflict between coercion-resistance and end-to-end verifiability.

2.1. Security requirements for elections

Each election is unique due to the election type, political landscape, legislation, cultural aspects of the society, available technology, used voting systems, threat actors, and risks. In addition, operating procedures play a significant role by defining how the election-specific activities have to be conducted. These also involve human interaction with technology. Thus, the security requirements for holding elections have to be adapted to the specific environment.

The means how the security requirements are fulfilled can vary. While voters can do some checks, there can be tasks assigned to the election observers and auditors. Next, we give an overview of the common security requirements.

We provide informal descriptions of the security requirements, which have been compiled based on the properties and security features commonly stated in the scientific literature [275, 140, 21, 149, 283, 314, 108, 168, 232, 54]. However,

we do not aim to list all of the security requirements and mainly focus on the requirements that are relevant for electronic voting systems. In addition, as some of the requirements are contradictory and depend on the election-specific context, it is not expected that all election systems fulfil all of the security requirements. The following list of requirements is provided in alphabetical order.

Accessibility. The voting system must be usable by voters who have disabilities.

Accountability. If a voting protocol participant misbehaves, it must be possible to correctly determine who is responsible and hold that participant accountable to resolve the dispute [201].

Availability. The voting system must be available to the voters during the elections. This includes both the client-side voting application and the back-end system. The voting system has to be fault-tolerant and able to handle unexpected peaks in the workload.

Ballot secrecy / vote privacy. The security measures must ensure that the voter can cast a vote so that its contents remain secret. The security measures must prevent the information on the filled-in ballots from being linked to voters' identities. Some definitions of ballot secrecy also include receipt-freeness and coercion-resistance, but we use these terms separately in this thesis. While the term ballot secrecy is more common, the term vote privacy is sometimes used in the scientific literature to carry the same meaning. More broadly, ballot secrecy can be seen as means for guaranteeing free and fair elections [217]. In principle, the information on the ballot can also be published in case the i-voting system guarantees that the ballot can not be associated with the identity of the voter who filled it in.

Cast-as-intended verification. The voter must be able to verify that their will is correctly encoded in the ballot sent to the voting system.

Coercion-resistance. The security measures must prevent a coercer from succeeding in coercing the voter. Voters must have the means to avoid complying with requests to cast a vote according to the instructions of the coercer. In this thesis, vote-buying is considered coercive behaviour.

Confidentiality of partial tally. A partial tally must not be published while voters are still able to vote.

Correctness. The election result must be correctly determined based on the valid votes cast by eligible voters.

Directness. The voter must cast a vote themself without delegating the right to cast a vote to a third party.

Dispute resolution. There must be a mechanism to resolve disputes. For example, in the case of electronic voting, it must be possible to report irregularities to detect both bugs and malicious behaviour. When an error occurs during vote verification, the voter may face a conflict between reporting the details of the issue and the secrecy of the cast vote. The election organiser also faces a dilemma, as, without strong proof, they cannot determine whether the voter is being honest or

trying to issue a false statement. Thus, creating a dispute resolution protocol can be a non-trivial task that has to be tailored for the specific voting protocol.

Eligibility. Only eligible voters must be able to participate in the elections. One way to enforce this requirement is to maintain a list of eligible voters and ensure that non-eligible voters can not cast a vote. However, there must also be a set of rules on how to update that list. For example, when a person has a birthday during the elections, which makes the person eligible to vote, should the list of eligible voters be modified? The same question holds for people who die during the elections.[1] In addition, it should not be possible to add ineligible persons to the list of eligible voters. However, the election organiser decides the method for determining eligibility unless it is regulated by law. As eligibility can be determined by checking voters' identities, voters should be provided with the means for proving their identities.

Eligibility verifiability. Anyone must be able to verify that the ballots that reached the tally originated from eligible voters. This requirement is applied to electronic voting systems.

End-to-end verifiability. Both the voters and the general public must be able to verify that the election result was determined correctly. A voting system is end-to-end verifiable when it provides cast-as-intended verifiability, recorded-as-cast verifiability, and tallied-as-recorded verifiability. According to some definitions, it is also essential to have eligibility verifiability to guarantee that only votes from eligible voters are tallied. For example, individual verifiability and universal verifiability may not be sufficient to provide end-to-end verifiability [202]. In addition, end-to-end verifiability can not be achieved in case individual verification is susceptible to clash attacks [203, 97]. Although the guarantees provided by cast-as-intended verifiability, recorded-as-cast verifiability, and tallied-as-recorded verifiability may not always be sufficient to guarantee end-to-end verifiability, they form the basis for achieving that.

Generality. All eligible voters must have the right to express their will via participating in elections. One's social status must not affect the possibility to vote. Thus, the means to vote should be accessible to voters regardless of their wealth.

Individual verifiability. Voters must be able to check that their ballots were correctly encoded and correctly recorded by the voting system. Individual verifiability is represented by the following two verifiability components: cast-as-intended and recorded-as-cast.

Over-the-shoulder coercion-resistance. A voter must have the means to counter coercion even if the coercer is physically present during the vote casting process [74]. However, it is assumed that a coercer can not monitor the voter for the entire voting period. Voting systems that follow this property either allow the voter to re-vote or provide the voter with an option to cast fake votes that are not

[1] To prevent fraud and misuse of voting credentials.

included in the tally.

Recorded-as-cast verification. The voter must be able to verify that the voting system correctly recorded their vote.

Receipt-freeness. The voter must not be able to provide strong proof about the vote that was cast. More specifically, the voter must not be able to prove that their vote for the particular candidate was included in the tally. The latter is in direct conflict with the requirement of end-to-end verifiability [98], which provides the voter with an option to check whether their vote was included in the tally.

Reliability. There must be a degree to which the voting system functions in an expected and specified manner during the elections.

Robustness. A voting system must be able to resist attacks and faults. The level of robustness illustrates how well the voting system adapts to attacks and faults [283].

Software independence. An error or change caused by software must not result in an undetectable change or error in the election result [275]. One approach to achieve this property is to rely on a paper-trail or paper-record, which can be used to detect irregularities. Some electronic voting systems use cryptography to provide verifiability, making it possible to detect errors or changes.

Tallied-as-recorded verification. Everyone must be able to verify that all correctly recorded valid ballots are included in the tally. In addition, it must be possible to verify that the tallied votes originate from eligible voters. Thus, the public can check that the election result is correct and represents the votes cast by eligible voters.

Tally integrity. The voting system must guarantee that each valid vote is included in the tally.

Transparency. It must be made possible to audit and observe the procedures related to elections. It must be possible to audit the software that is used to conduct elections.

Uniformity. Every voter must have an equal opportunity to affect the outcome of the elections through the voting process. This means that all eligible voters must have an equal amount of votes, and all the votes must have the same weight in determining the election result.

Universal verifiability. It must be possible for anyone to verify that the election result is correctly calculated from valid ballots. Sometimes universal verifiability is also considered to include eligibility verifiability.

Usability. The voter must be able to understand what has to be done to cast a vote so that the voter can take part in the elections. In the case of electronic voting, the voting software must be easily usable. Voters must not face ambiguity when using the voting software. In addition, voting systems must be made accessible for voters with disabilities.

Verifiability. Voters must not have to solely rely on trusting the election results. They must also have the means to check whether the processes work as intended.

These means are commonly represented with three verifiability components: cast-as-intended, recorded-as-cast, tallied-as-recorded. The first two represent individual verifiability by allowing voters to check that their votes were correctly encoded and their ballots correctly recorded by the voting system. Tallied-as-recorded represents universal verifiability, which allows anyone to verify that the election results are correct. An end-to-end verifiable voting system fulfils all of the three aforementioned verification requirements.

2.2. Overview of technologies and methods for protecting ballot secrecy and integrity

In the following, an overview is given of technologies commonly used by online voting systems. However, we do not aim to list all cryptographic technologies that could be used in election systems.

End-to-end encryption. The secrecy of the ballot has to be protected while it is being transmitted from the voter to the voting system and from the voting system to the tallying system. This can be achieved with the help of end-to-end encryption (E2EE). A straightforward approach is to generate an election-specific key pair for the chosen asymmetric cryptosystem and make the public key available to the voting software. The voting software uses the election public key to encrypt the ballot to protect its contents during delivery. The corresponding private key is used to decrypt the votes so that they can be counted. However, to protect ballot secrecy, it is critical to ensure that the private key is protected throughout its lifetime, including generation, storage, distribution, and destruction. In addition, it must not be possible to use the private key to decrypt votes while the elections are ongoing or when the encrypted votes are linked to identities.

Digital signatures. Voters can protect the integrity of the ballots by digitally signing them. The signatures issued by the voters may be linked to identifying information that can be used to verify the eligibility of these voters. In the context of legally binding elections, it is critical to use both a secure signing scheme and a secure method for managing the signing keys.

Secret sharing. The decryption key used to open the ballots has to be secured to protect ballot secrecy. One way to prevent the decryption key from being leaked is to use secret sharing. Secret sharing was introduced by Shamir in 1979 and provided the means to distribute a secret between a selected number of parties [286]. For example, the private key could be secret-shared to be recombined only when a threshold number of keyholders are present. By choosing trustworthy keyholders, it is possible to lower the probability of private key leakage. However, if a threshold number of keyholders collude, it may be possible to violate ballot secrecy.

Re-voting. By allowing the voter to change the vote, both coercion and successful vote-buying become more challenging. When a vote is cast under coercion, the voter can re-vote later and invalidate the previous vote. However, this measure gives malware additional means to interfere with the elections. In case

malware has access to voter's voting credentials, it could invisibly re-vote and thereby invalidate the previously given vote. A voter may not be able to verify whether the voting system recorded the vote given by the voter or a re-vote by malware. If the voter would be allowed to verify the vote until the end of the voting period, the verification mechanism could be used as proof for the coercer or vote buyer. However, when verification would be limited to a certain time period, malware could safely re-vote without the voter noticing through the verification system that the initial vote was overwritten.

Code voting. Remote online voting has an inherent problem with vote integrity and ballot secrecy due to the untrustworthiness of a general-purpose computer. When the computer used to cast a vote is compromised, it endangers not only voter's privacy but also vote integrity. Chaum proposed to solve these issues by using code voting, which works by pre-distributing unique codesheets to the voters that map codes to candidates [67]. The codes on the codesheet must be kept secret and must not be stored in the computer that is used to cast a vote. In case the compromised computer does not have access to the mapping between candidates and codes, it cannot violate ballot secrecy. Also, malware on the compromised computer would also not be able to cast a vote for the preferred candidate as it would not have access to the corresponding code. However, the difficulty of this approach lies in the logistics of securely printing and delivering the codesheets to the voters in a manner that keeps the binding between the codesheets and voters private. The complexity of such a logistical system was illustrated in Norway in 2011, where during the printing of poll cards, the binding between voters and the poll cards were mixed up in some cases [296]. Furthermore, usability aspects have to be considered as voters would not be able to directly select or mark the candidate's name. In addition to the issues mentioned above, the requirement to pre-distribute physical items means that such a voting system can not engage remote voters who are travelling or are located in hard-to-reach locations. While using a local postal service to deliver code sheets might work, it becomes more challenging to guarantee reliability once voters rely on international postal services. In addition, when using such a delivery method, the security and trustworthiness of the postal service become paramount.

Return codes. There are two main approaches that have been proposed for creating cast-as-intended verifiability in the context of i-voting [185]. When the ballot is encrypted, it is either possible to verify that the correct vote was encrypted or that the decryption of the encrypted ballot returns the correct vote. The former can be implemented via the Benaloh challenge [47] or by repeating the encryption of the ballot on an independent device to compare the ciphertexts. The latter can be implemented by allowing the verification device to open the vote, which can be done by sharing the randomness used for ElGamal encryption. Another option is to use return codes. In this case, a random-looking code is returned to the voter after casting a ballot. The voter can compare the return code with a pre-distributed codesheet delivered over an independent communication channel.

Distribution of trust. In an ideal setting, it should be possible to audit all the processes that could affect the integrity of the election outcome or the voters' privacy. However, only a tiny fraction of voters are auditing the election processes. Thus, some parts of the election systems must be trusted by the voters to behave as expected. However, the trust assumptions can be somewhat reduced by distributing the critical tasks between multiple parties. For example, Feldman's secret sharing scheme can be used to generate an ElGamal key in a distributed manner so that only at least a threshold number of participants can reconstruct the secret key [122]. The parts of the voting infrastructure that have to be trusted likely represent single points of failure of the system. Thereby, such components or roles could be distributed between multiple stakeholders. For example, when the voting client software would be vendor-independent and open-source, the voters could choose which implementation to trust.

Re-encryption mix-nets. David Chaum introduced the concept of a mix-net in 1981 [66]. He described how input messages could be mixed so that the parties receiving the output messages would be unable to link them back to the inputs. This can be achieved by both re-encrypting and permuting the input ciphertexts. In the context of voting, re-encryption mix-nets are often implemented using the ElGamal cryptosystem due to its homomorphic property. Thereby, it is possible to re-encrypt the ballots without affecting the integrity of the votes on the ballots. Thus, re-encryption mix-nets can be used to hide the connections between encrypted votes and voters who cast these votes. One of the most well-known examples is the Verificatum mix-net [319], which has been used in national elections in Norway, Estonia and Switzerland [153].

Ring signatures. Rivest, Shamir, and Tauman introduced the concept of ring signatures in 2001 [274]. Ring signatures make it possible to sign on behalf of a chosen set of ring members without revealing which member produced the signature. Contrarily to group signatures, there is no manager, and the signer's anonymity can not be revoked. Eos is a voting scheme that is built on top of ring signatures [255].

Zero-knowledge proofs. It is non-trivial to check the integrity of processes used in an election system without breaking any privacy requirements. However, advanced cryptographic concepts like zero-knowledge can provide answers to some of these conflicting requirements. Goldwasser, Micali and Rackoff first published the concept of zero-knowledge in 1985 [141]. Zero-knowledge proofs make it possible to prove a claim without revealing any auxiliary information about it. For example, such proofs are commonly used by i-voting schemes to guarantee that the ballot was filled with a valid value without revealing the vote itself. In addition, zero-knowledge proofs can be used to prove the correctness of decryption without revealing the decryption key.

Fake credentials. One of the countermeasures against coercion is to use fake credentials while casting a vote [181]. This gives the coerced voter a chance to cast an invalid vote or notify the election officials that the vote was cast under coercion.

However, the necessity to manage fake credentials can reduce the usability of the voting system.

Anonymous communication. There are i-voting schemes that provide participation privacy by allowing voters to submit ballots over an anonymous communication channel. Such schemes usually rely on Tor to hide the IP address of the voter.

Verifiable decryption. To guarantee that decryption was done correctly, it can be proven that the decrypted information was calculated from the given cryptogram [61]. In the context of voting, verifiable decryption allows auditing of the decryption process without revealing the decryption key. Thus, third parties can check that the election result was calculated from the encrypted ballots that were sent to be decrypted and counted.

2.3. The role of encryption in ballot secrecy

It is important to highlight the threats that could impact a large part of the electorate. One such threat is the possibility of votes being linked back to voters after the elections are over.[2] If this threat is not mitigated, it is possible to coerce voters before the elections and persecute them later based on how they voted.

In the case of electronic voting systems, ballot secrecy is often guaranteed by encryption. However, the commonly used cryptosystems are not designed to provide everlasting security due to the computational assumptions [310]. Encrypted ballots could be endangered by breakthroughs in cryptanalysis, computer hardware, weaknesses found in algorithms, advancement of quantum computing, usage of poor randomness, and the leakage of the corresponding secret key.

Different estimates are given regarding the safety of commonly used cryptosystems and key lengths. For example, NIST's Recommendation for Key Management, published in 2020, states that for symmetric encryption, 112 bits of security is sufficient until 2030, while the NSA states that 256 bits are already required since 2015 [42, 17]. The estimates for the minimum required key sizes also vary for public-key cryptosystems [11, 42]. Thus, without relying on everlasting privacy, long time storage of encrypted ballots would be risky due to the uncertainty of how long ballot secrecy would last.

The long-term risks related to encryption are one reason why digital ballots are destroyed after complaints have been reviewed and the legally set retention time has expired [6]. However, the requirement to destroy ballots after a legally set retention period is common practice also for paper-based voting systems [241, 240, 1]. This provides an opportunity for timely legal disputes while preventing endless challenges about the election results.

[2]The possibility to link votes to voters does not automatically imply that the functionality is abused. For example, in the UK, the back of the ballot paper contains a serial number that can be used to link the vote to a voter [269]. Although that option is rarely used, OSCE/ODIHR recommended removing the numbering from the ballot to protect voters' privacy [128].

However, it is important to note that electronic voting differs from paper voting because the destruction of ballots does not provide a guarantee that copies of the encrypted ballots do not exist.

In the case of remote electronic voting, voters have to send encrypted votes to the voting system. The communication between the voting software and the server of the voting system can be secured with the help of transport layer security (TLS). If the server's certificate is pinned or mutually authenticated TLS channel (TLS-CCA) is used, man-in-the-middle attacks become very difficult to conduct [252]. Thus, we can assume that the traffic protected by a mutually authenticated TLS channel can only be read by having access to the communication endpoints or the corresponding private keys. To prevent the vote collector server from violating ballot secrecy, the ballots must be separately encrypted with the election-specific public key before they are sent through the TLS channel.

The usage of TLS does not prevent third parties from capturing encrypted traffic. Thus, it becomes relevant whether something useful can be done with the captured traffic. If the voting servers require forward secrecy for TLS connections, each communication session is negotiated with fresh keys that are not derived from the server's long-term private key. Thus, forward secrecy protects the previously negotiated session if the server's private key gets compromised. However, current TLS versions do not provide post-quantum security.

2.4. Threat actors

For analysing the security of voting systems, the threats and threat actors have to be identified. As the dissertation focuses mainly on the client-side of voting systems, we cover the threats related to coercion and vote integrity. Thus, the threats that are relevant only to the server-side are considered out of scope.

In general, the threat actors could be classified by their influence. However, influence is hard to measure as it can be applied indirectly. For example, it is difficult to distinguish legitimate political advertisements from malicious activities, which may fall under the scope of information warfare. When considering coercion, we mostly ignore the indirect impact and focus on the threat actors who have at least some level of control over voters or their devices.

We assume that the aim of an attacker is to influence or change the election result. Therefore, we could compare the threat actors based on the number of voters that they could target.

The least impactful of such threat actors is a single person who targets family members. For example, family members could be pressured or intimidated to vote in a specific way. It is questionable whether technical measures that aim to provide coercion-resistance would be effective in such a situation. The social aspects of coercion-resistance should be studied further.

A more impactful threat actor may have an influential position in a small organisation. In some cases, employers can coerce their employees. However, also

other dependent parties may be vulnerable to coercion. For example, this may hold for nursing home employees who can have a significant impact on the people staying there.

Large corporations or influential people employed there could also be considered a class of threat actors due to their resources, allowing them to influence politics and technology used in elections. Their resources do not have to be restricted to money, as software, services, and data play a significant role in the modern interconnected world. For example, consumer data can be repurposed to profile and influence the voters. In addition, election systems that rely on computers are at least partially dependent on the software vendors. Furthermore, some service providers like certificate authorities and authentication providers can be critical for the election system to function correctly.

In addition, political parties and politicians have the motivation to get elected. Not all of them behave within the limits of the law. Thus, corrupted politicians could also be considered as threat actors. Furthermore, if corrupt politicians, corrupt officials, or corrupt election administrators can control the election processes, they may be able to affect the election result. This is one of the reasons why many i-voting schemes attempt to minimise the amount of trust that is placed on the election organisers [15, 75].

A threat actor, which has access to a significant amount of resources, is a foreign state. Such resources and capabilities could be used to influence both the politics and elections of a competing or adversarial state. However, the interference does not have to be executed directly, as it is possible to outsource the task to friendly groups of cybercriminals. While cybercriminals who act alone could also be seen as threat actors, they are unlikely to be motivated to attack an election system due to the lack of financial incentives. In addition, as election interference is a high-profile attack, it is likely to be investigated thoroughly, raising the chances of an attacker getting caught. Therefore, if cybercriminals are involved in targeting elections, we assume they cooperate with a state-based threat actor.

2.5. How does the voting environment influence voters' privacy?

While technology and processes can protect the secrecy of most ballots, it is very difficult to guarantee that for every ballot regardless of the voting method.

In the case of paper voting, voters could be forced to write unique markings on the ballots so that malicious election observers could identify them. The same holds for some electronic voting systems, which allow the voters to use write-in fields [195]. Furthermore, the polling booth itself is not guaranteed to provide a private vote casting environment, both due to audio-based side-channels [197] and voters being able to use small recording devices to capture the vote casting process [48]. While this could be prevented by searching the voters, it would be a significant intrusion of voters' privacy, which could discourage voters from

voting.

Information could also leak via the electronic services used by the voting system. A good example is OCSP queries, which are used to check the validity of X.509 certificates. If an electronic voting system checks the validity of voters' certificates, the OCSP service sees which voters participated in the elections.

Remote electronic voting and postal voting introduce new threats due to the uncertainties originating from the vote casting environment. The lack of a voting booth makes it easier to monitor and influence the voter. This can be a significant issue in the context of family voting. A potentially more severe issue is related to the vote casting software running in a general-purpose computer. The computer where the voting software is running can be considered an untrusted environment, which may contain malicious software [24].

If the voting device is infected with malware, it can no longer be trusted to behave as expected. Thus, a malicious application could intercept, modify, and leak the information inserted or selected by the voter [164, 239]. Such a threat is significant due to the possible amount of voters who could be affected.

Verification mechanisms provided by the voting systems are a mitigation measure designed to detect attacks that may change the outcome of the elections. However, in general, these procedures cannot identify malware that only collects information about voters' choices without interfering in the vote casting process. Thus, verification does not aim to solve issues with vote privacy in the context of untrusted end-user devices.

In principle, there are three approaches to solving the privacy issue mentioned above. The first option is to eliminate the risk so that the end-user devices would not be vulnerable to malware. However, we do not know how to achieve that. The second option is to cast a vote in a secure computing device specially designed for elections. That approach may work, but it is likely to be prohibitively expensive. The third option is to use an independent communication channel to deliver the voter a codesheet containing candidate encodings, which the malware cannot link to the candidates' names. Thus, the voter would have to enter an encoding of the candidate while voting. However, implementing code-voting for general elections is challenging due to financial, usability, and logistics challenges.

2.6. How to guarantee election integrity?

When assessing the integrity of elections, all of the subprocesses have to be analysed. These involve managing the list of eligible voters and the list of candidates, distributing the ballots, authenticating the voters, keeping a list of voters to guarantee each voter gets one vote, observing the ballot boxes in the polling stations, monitoring / observing the voting process, transporting the ballots or ballot boxes, storing the ballots or ballot boxes, counting the ballots, delivering the results, auditing.

Paper-based voting systems usually leave most of these processes monitored

either by independent observers or officials who help conduct the elections. In such systems, an individual voter can not control how his vote is processed or how the rest of the votes are handled. This task is delegated to the observers and officials who have to be trusted to behave honestly. However, the voluntary monitoring system that is based on election observers can function efficiently only if certain preconditions are fulfilled.

It is essential that there would be a sufficient number of observers as otherwise, it is difficult to cover the whole election lifecycle. In addition, the election observers have to be aware of the risks and should have the means to report the misconduct so that the findings would be taken into account. The latter relies on an independent justice system, free media, and election organisers that are politically unbiased. Thus, it is difficult to hold fair elections in a society where there are limited means of holding corrupt entities accountable. Therefore, all of the former should be considered a precondition for guaranteeing the integrity of the elections.

However, some voting channels are more difficult to audit and observe than others. For example, in the case of postal voting, the vote casting process and the ballot delivery can not be observed by volunteers. Thus, the risks related to postal voting have to be studied. For example, this has been done in Switzerland [187]. Furthermore, in some cases, the delivery of traditional ballot papers also depends on the postal service [110].

Compared to traditional paper-based voting systems, it is relatively easy to make electronic voting systems verifiable. Thereby, some trust assumptions can be reduced by giving voters the responsibility to verify their votes and report any inconsistencies.

There are also partially paper-based voting systems that rely on cryptographic methods for verifiability [281, 68]. However, it is difficult to use such voting systems in large scale elections as cryptographic material would have to be securely printed and distributed, which creates a security-critical logistical problem [136].

In some cases, cryptographic checks can be added to existing voting systems so that they can be used in parallel with the existing systems. This approach is taken by Microsoft's ElectionGuard software development kit [220], which was released in 2019. It provides open-source software and specifications that make it possible to run verifiable elections. As ElectionGuard's API can be integrated into existing election systems, it is possible to add verifiability to voting systems that rely on electronic ballot marking devices.

ElectionGuard provides means to produce a digital tally so that voters can verify both their votes and the tally's correctness. Hence, voters are given tracking codes while casting a vote. To mitigate the threat of a malicious voting client, the voter can audit the encrypted ballot with the help of the Benaloh challenge [47]. When considering the integrity of the tally, ElectionGuard provides strong cryptographic guarantees, which makes large scale election interference detectable. However, it does not mitigate the threat of coercion as the tracking code used for

verification can also be used to prove how a voter voted.

2.7. Finding a balance between conflicting requirements

The notion of security depends on the context. Differences in requirements and levels of accepted risk make it challenging to create a universally accepted and secure voting system.

If the vote is cast in an uncontrolled environment, the voting system can only provide mitigation measures that reduce the risk of coercion. By providing the voters with coercion-resistance measures, it is attempted to make it impossible for the coercer to check how the vote was cast.

End-to-end verifiability provides the voter with the means to check how the vote was cast. This verification method could also be used by a coercer as the vote can be verified in an uncontrolled environment. Thus, when designing an online voting system, a balance has to be found between the integrity and coercion properties. Doing this is neither straightforward nor universal.

It is common for end-to-end verifiable voting systems to publish cryptograms to a bulletin board along with the cryptographic information required for verification. This allows voters to verify their votes, but it also makes it possible to remotely observe the elections to detect anomalies. However, publishing such information on a bulletin board would introduce new risks. For example, the cryptosystems that are commonly used to encrypt the ballots do not provide everlasting privacy and are not secure against powerful quantum computers.

The debate regarding the privacy and verifiability aspects of voting systems is no longer confined to academic literature. Until recently, the possibility to verify cast votes was implemented only in a few i-voting systems. However, technologies like Microsoft's ElectionGuard have demonstrated that vote verification can be integrated into existing election systems.

Thus, the discussion about the conflict between verifiability and coercion-resistance is likely to expand to existing election systems and thereby become more prominent. In addition, the increased polarisation of societies, the existence of echo chambers, and the spread of fake news along with conspiracy theories highlight the necessity to transparently prove that the election results have not been tampered with [148, 316, 20, 210, 257, 69, 41]. Trust in the election systems can be increased by allowing voters to verify that their votes were taken into account and that the election result was calculated correctly. Such an outcome might have a positive effect on the health of democracy. However, this can only be achieved when voters trust the verification mechanisms.

In principle, conspiracy theories can be created to undermine the verification system. Thus, the verification system should be sufficiently simple so that voters can be convinced that they can check the election outcome. Therefore, it is not sufficient to just implement end-to-end verification or conduct risk-limiting audits [211] to gain voters' trust. Also, the principles behind the verification mech-

anisms and post-election audits should be explained to the voters. This requirement means that the designers of the verification system also have to consider non-technical aspects like complexity, understandability, and explainability.

As it is not practically feasible to explain complex cryptography to an average voter, the voting system should be made as transparent as possible. For example, voters could be given access to open-source tools to verify that the election result is calculated correctly. The trustworthiness of such tools could be increased by publishing the source code along with detailed documentation, manuals, and reasoning regarding the design choices. In order to make auditing of the source code more manageable, the implementation could reflect the pseudo-code documentation as illustrated by the CHVote project [152]. That would allow third parties to audit and cross-reference the cryptographic protocols with the implementation. Nevertheless, the trust assumptions can not be completely eliminated. However, the same holds for the existing paper-based voting systems that do not incorporate end-to-end verifiability.

In general, any type of vote verification is likely accompanied by threats related to coercion and vote privacy. However, not all coercive behaviour has the same effect. For example, if the coercer aims to change the election result and can freely pick targets, the most efficient option would be to force voters from the opposing side to vote for a candidate selected by the coercer. While the coercer can also force voters to abstain, it requires twice the amount of voters to be targeted to achieve the same result. In addition, when coercive behaviour is not coordinated and affects all parties, the changed votes are likely to be dispersed between candidates and parties, making it more challenging to shift a sufficient amount of votes to change the election outcome. Family voting is one example of such coercive behaviour. When leaving the motivation of coercion aside, the leakage of one's vote may have long-lasting implications for the voter.

The advancement of technology has made it practically impossible to guarantee absolute ballot secrecy for every voter, which means that some coercion risks have to be accepted. However, academic literature tends to use terms like coercion-resistance, which does not perfectly reflect the reality where limited use of coercion is possible and can not be avoided. To overcome this issue, Matthew Bernhard introduced the term coercion-hardness to describe a voting scheme, which ensures that changing the election outcome via coercion is difficult [53]. He tied the definition of the term to the margin of the election.

By having a numeric estimate for the election margin, it would be possible to state the level of required coercion-hardness for a voting scheme. Such metrics could be used to determine the acceptable level of verifiability provided to the voters without risking the election outcome being changed. It is apparent that the margin of the elections can not be precisely determined in advance. However, opinion polls can give an estimate, which could determine a probable range for the margin. The rest would be left to the decision of fixing an acceptable risk level.

Determining the level of accepted risk is non-trivial as it does not only depend on the risk of coercion changing the election result but also on the risk of the election result being changed due to the lack of verifiability. One way to find the balance between coercion-resistance and verifiability is to estimate the probabilities of an election result being changed by different types of attacks. However, it is difficult to estimate how coercion could impact voters' lives.

As the risk to the integrity of the elections is not guaranteed to be constant, the risk levels would have to be determined before each election. Thus, depending on the outcome of the risk analysis for the specific elections, the security measures would have to be re-balanced based on the level of risk. While that may seem doable from a theoretical viewpoint, applying such mitigation measures in practice would be difficult. First, it may be difficult to obtain reliable data for the risk analysis. Second, it would require a voting scheme that is easily adjustable according to the level of risk. Third, both the public and the politicians would have to trust the risk analysis and the people behind that. Fourth, both the politicians and voters are not accustomed to a dynamic voting system that constantly changes.

3. HOW TO ACHIEVE COERCION-RESISTANCE IN THE REMOTE SETTING?

In a democratic society, every eligible voter should have the opportunity to vote. However, the changes in society have made voting at a polling station inconvenient or impossible for many people. For example, the last decades have shown an increase in people who work abroad [309]. At the same time, digital technologies have made the life of people with disabilities easier. In addition, the COVID-19 pandemic showed that an infectious disease could significantly affect how elections are organised and conducted [205, 125]. The above shows that alternative voting methods have to be introduced to prevent many eligible voters from being effectively disqualified [193, 238].

One approach to solving such problems is to allow the voter to cast a remote vote. For example, this can be done via mail, proxy voting, or by asking the election officials to make a home visit with the ballot box. However, these methods have issues. Doing home visits does not scale well, carries the risk of infection during a pandemic, and is expensive for voters who live in remote areas or abroad. In the case of postal voting, it is difficult to authenticate the voters. Due to postal service disruptions, some votes may get lost or arrive too late [295, 171]. There can also be legal issues with the cross-border delivery of voting documents [226]. In addition, some voters may be vulnerable to coercion as there is no protective polling booth in the remote setting.

The aforementioned problems are amplified due to the spread of disinformation. As voters can not directly verify that postal voting does not contain inconsistencies, it is practically impossible to refute all claims regarding fraud. Regardless of its issues, postal voting is used all over the globe [187, 267, 226]. One of the reasons for that is the lack of a suitable replacement.

In general, there are two main ways to mitigate the issues with postal voting. First, postal voting could be improved. For example, cryptography could be used to add verifiability to existing systems [49, 101]. Second, postal voting could be replaced by a different remote voting channel. The research and practical experiences from the past two decades have shown that remote electronic voting can remedy some of the issues related to the delivery of ballots, the integrity of ballots, and the disenfranchisement of voters.

While remote electronic voting can provide guarantees regarding the delivery of ballots, there is no straightforward mitigation for coercion in the remote setting. Although different measures have been proposed in the scientific literature to counter the issue, there is a lack of practical experience in implementing and trying these measures. This is partially due to a limited number of countries experimenting or using remote voting solutions and partially due to how research is conducted. Researchers have few incentives to build and test prototypes, as this is usually not required to get a publication. Thus, research papers describing

new voting schemes often only contain an abstract overview of the measures for mitigating coercion, leaving the details to be figured out by the implementer or another researcher.

We reviewed a selected number of i-voting schemes that provide at least a partial level of coercion-resistance [195]. It turned out that by going into the details, cracks started to appear in the proposed mitigation measures. The reviewed anti-coercion measures turn out to be either inefficient against some types of coercion attempts or difficult to implement in practice. The difficulties with implementation are often caused by the assumptions that are made while designing a voting scheme. For example, the anti-coercion measures of voting schemes sometimes rely on hardware that is not readily available in the market. In addition, it is often assumed that voters are able to apply the mitigation measures proposed for the new voting scheme. However, it turns out that there is a lack of usability studies to confirm whether voters even understand how the anti-coercion mitigation measures work. Ironically, it is impossible to conduct the usability studies without at least a working prototype implementation of the voting scheme.

The next section describes how the meaning of coercion-resistance has evolved in the scientific literature. Section 3.2 gives a general overview of coercion-resistant voting schemes. The following sections describe the anti-coercion properties of the selected i-voting schemes studied in [195]. Finally, Section 3.10 summarises the general issues of achieving coercion-resistance.

3.1. How to define coercion-resistance?

The difficulties in determining the abilities of an attacker and the accepted level of coercion affect how coercion-resistance is defined in scientific literature. In the nineties, coercion-resistance was mostly covered in the form of receipt-freeness. This is illustrated by the number of definitions describing receipt-freeness [51, 246, 218, 179, 108, 158, 65].

However, receipt-freeness does not cover all types of coercive attacks. Thus, the definition for coercion-resistance should be sufficiently abstract to cover all coercive actions that could influence the election outcome. As there is no universally accepted definition to describe coercion and coercion-resistance [155], we decided to use a coercion-resistance definition, which covers a broad range of attacks.

Such an approach comes from Juels *et al.* [181], who stated that a scheme is coercion-resistant if it is infeasible for an attacker to find out whether a voter complied with the request of the coercer. In addition, they stated that a coercion-resistant voting scheme should prevent the following three coercion strategies:

- forcing the voter to abstain from elections,
- forcing the voter to cast an invalid vote,[1]

[1] Juels *et al.* used the term randomization attack [181].

- using voter's credentials to cast a valid vote.[2]

If we assume that an adversary can continuously monitor the voter from the registration phase until the voting ends, coercion-resistance in the remote setting becomes practically impossible to achieve. However, such invasive attacks are unlikely to scale well. Thus, it is more realistic to analyse coercion in the context of subprocesses relevant to elections.

Benaloh [48] describes coercion in three temporal settings: before voter registration, between registration and voting, and after voting. It is important to note that these settings set different limits for the adversary. For example, it is difficult to prevent coercion when the voter can be coerced before the registration as the coercer may get access to valid voting credentials. In case coercion is attempted after voter registration, but before the vote has been cast, the success of a coercion attack depends on the attacker's capabilities and the anti-coercion properties of the voting scheme. However, if coercion is attempted after the vote has already been cast, a coercion-resistant voting scheme should prevent both the attacker and the voter from being able to prove how the vote was cast. Thereby the scheme would protect vote privacy. Thus, the critical part in defining coercion-resistance lies in the assumptions that limit the capabilities of an adversary.

Some voting schemes are designed to be coercion-resistant against a passive coercer [200]. This type of adversary is not actively monitoring the voter but aims to buy votes or intimidate voters into voting according to the provided instructions. Some other voting schemes also attempt to mitigate active coercion where the coercer directly interferes with the voting process [181, 255].

It turns out that the issues of voluntary vote selling and coercion have much in common [48]. For example, if a vote seller has the means to prove how a vote was cast, the same method could be used to coerce the voter. In general, a coercion-resistant voting scheme has to prevent the coercer from finding out how a voter voted even if the voter is willing to cooperate with the coercer. For systems that allow voters to re-vote, the coercer should not be able to determine which vote was cast last.

In some cases, it may be necessary to prevent an adversary from finding out whether specific voters have voted. If such information leaks, a coercer could confront the coerced voters to check whether they followed the instructions to abstain or not to re-vote. Different techniques have been proposed in the scientific literature to provide participation privacy to the voters. For example, voting systems can hide the information about voter participation by providing means for the ballot to be delivered over an anonymisation network like Tor [74, 231], allowing dummy votes to be cast in voter's name [200], and signing ballots with ring signatures [255].

Ballot secrecy forms the cornerstone of coercion-resistance. When the voter is

[2] Juels *et al.* used the term simulation attack to describe an attacker accessing voters' credentials to vote on their behalf [181].

confident that the vote remains secret, it becomes difficult to intimidate and coerce the voter. However, it turns out that basic ballot secrecy is not sufficient to protect the voter in case the voter can be forced to prove how a ballot was filled.

Some voting schemes provide receipts to voters to enable vote verification and thereby increase the auditability of the corresponding voting scheme. The receipt or the method used for verifying the vote can also be used as leverage by the coercer. In the straightforward case, the receipt leads to cryptographic proof, which binds the voter and the cast vote [165]. Thus, the voting schemes that attempt to simultaneously achieve verifiability and coercion-resistance have to either rely on extra assumptions or use complex anti-coercion measures to reduce the threat of coercion. In addition, Chevallier-Mames *et al.* have shown that universal verifiability and receipt-freeness can not be achieved simultaneously unless private channels are available between voters and voting authorities [70]. The same authors also proved that universal verifiability and unconditional privacy can not be achieved simultaneously unless all registered voters participate in the elections.

The voting schemes, which do not give the voter means for providing strong proof of how the vote was cast, are said to be receipt-free. While the concept of receipt-freeness is easily comprehensible, applying it in practice is not straightforward as voters may also have indirect means to reveal how they voted. For example, a video recording displaying the vote casting process can be sufficient proof if re-voting is not enabled [48]. Even when a voting scheme is designed not to produce a receipt, it does not always guarantee that a voter can not prove how a vote was cast. Sometimes the designers of the voting schemes do not consider all possible attacks, as it happened with the Benaloh-Tuinstra scheme [51], whose receipt-freeness was shown to be broken [172]. While receipt-freeness on its own cannot guarantee coercion-resistance against an active adversary, the reverse holds, as it has been proven that coercion-resistance implies receipt-freeness [107].

One way how a voting scheme could reduce the impact of coercion is by limiting the time period when it is possible to access the receipts. Another easily usable mitigation is to allow voters to cast a re-vote, thereby invalidating the previous receipt. However, as a negative side-effect, the possibility to re-vote can weaken verifiability. In addition, re-voting does not provide a guarantee against all types of coercion. For example, adversaries could coerce voters at the end of the voting period to prevent the voters from re-voting. Unfortunately, such a type of coercion is difficult to mitigate. In addition, network-level monitoring might reveal whether multiple votes have been cast from the same IP address (unless anonymisation networks like Tor are used). To protect against such adversaries, the coerced voter would have to cast re-votes from different IP addresses. Alternatively, the voter can rely either on a trustworthy VPN service or Tor. Neither of the technology-based alternatives is an option for an average voter unless the functionality is integrated into the voting software. Even when the threat is not mitigated, it could be argued that access to network-level monitoring is available only to very powerful adversaries.

Another type of adversaries are the coercers who actively monitor the behaviour of the voter. Such an attacker may be physically present while the vote is being cast. Alternatively, an adversary might be able to monitor voter's computer. To counter active coercion, voters have to be able to falsely convince the malicious party that a vote was cast according to the provided instructions. There are multiple ways to do that, but these measures need conscious participation by the coerced voter. For example, some voting schemes rely on fake credentials or pseudo-identities to either cast fake votes or to covertly signal the election officials that a vote was cast under coercion [73, 255, 75]. However, the usability and effectiveness of such schemes need to be studied before similar measures can be applied to political elections.

The aforementioned issues, along with the possibility of recording or streaming the vote casting process, make it practically impossible to guarantee absolute coercion-resistance for every voter. In essence, coercion-resistance is a measure that is aimed to protect democracy by guaranteeing free and fair elections. To have fair elections, third parties must not be able to launch large scale coercion attacks that could change the election outcome. Therefore, voting systems should be designed to guarantee eligible voters the freedom to vote while limiting the possibilities for large scale coercion attacks.

Even when the voting scheme cannot protect every single voter from all types of coercion, the legislation usually provides some level of deterrence. While it may be possible to prevent small scale coercive attacks from being reported to the authorities, it is less likely that large scale attacks would remain hidden. Of course, it has to be understood that the penalties described in the legislation may not be an effective deterrence in a corrupt country where election-related crimes are not investigated.

3.2. Overview of coercion-resistant voting schemes

Many e-voting protocols that are either receipt-free or coercion-resistant have been proposed in the literature [51, 237, 282, 245, 246, 172, 45, 218, 224, 216, 75, 30, 74, 231, 22, 186, 200, 213, 65, 280, 255]. However, not all of them can be used in practice. For some of these voting schemes [51, 245, 218], the claims regarding receipt-freeness and coercion-resistance have been shown to be invalid [172, 246, 182], while some other schemes are not efficient for large-scale elections [181] or are not easily applicable in practice [231].

In general, it is common for the voting schemes to protect the voters' identities in the tallying phase either by relying on mix-nets [66, 282, 176, 15, 165, 266] or homomorphic encryption [76, 51, 100, 172, 105]. The same goal could also be achieved by using blind signatures [132, 246] or ring signatures [255], however, implementing these in practice is non-trivial. As we were interested in ways how coercion-resistance could be applied in practice, we selected some recently proposed voting schemes to be analysed in detail.

The following sections give an overview of the seven i-voting schemes that were studied in [195]. Out of the seven schemes NV-Civitas [231] and Selections [74] belong to the JCJ/Civitas family, while KTV-Helios [200] and BeleniosRF [65] belong to the Helios family. The other three are the Estonian i-voting scheme known as IVXV [165], Selene [280], and Eos [255]. All of the seven schemes provide basic ballot secrecy but use different strategies for mitigating coercion. The two main strategies for a voter to evade coercion are based on fake credentials and re-voting. The former gives an opportunity to cast a fake vote, which allows the voter to abide by the wishes of the coercer. The latter allows the voter to invalidate the vote that was cast while being under coercion. Table 1 gives an overview of the building blocks used by these schemes.

For each scheme studied in [195], we list the coercion-resistance properties and the assumptions that they are based on. These properties are summarised in Table 2. We describe the levels of coercion-resistance based on the definition from Juels *et al*. To make the comparison of the schemes more explicit, we added two categories, receipt-freeness and over-the-shoulder coercion-resistance [74]. The latter describes an attacker who can physically monitor the voter while the vote is being cast but can not monitor the voter for the whole voting period. While some of the studied schemes fulfil most of the coercion-resistance properties, they achieve that by relying on multiple assumptions. However, these assumptions may not hold in practice either due to the lack of supporting technology or due to the usability aspects. By identifying the weaknesses of the proposed schemes, we can move closer to designing voting schemes that balance the verifiability and privacy properties while being usable in practice.

3.3. Estonian i-voting scheme

Estonia started to use i-voting in 2005 [216]. Since its introduction, the i-voting scheme has had multiple adjustments. For example, individual verification was added in 2013. The functionality was provided via a smartphone application that allowed voters to check whether their vote was cast-as-intended and recorded-as-cast [169]. A major change to the voting scheme was introduced in 2017 by making the server-side processes verifiable [165]. The updated voting scheme was named IVXV. It is the scheme that we analyse in this section. This section provides a brief overview of IVXV, with the scheme depicted in Figure 1 and some of the main properties listed in Table 1. A more detailed overview of IVXV can be found in Section 5.1.

Before elections, the election organisers generate an election-specific keypair.[3] The public key is bundled with the voting client, and the private key is secret-shared between a selected number of key managers [89].

From the voter's perspective, the voting system works in a similar manner as

[3] Since 2017, 3072-bit ElGamal keys are used.

Table 1. The table gives an overview of the schemes studied in [195]. The lower part of the table contains the categorisation of the means used for protecting the identities of voters in the tallying phase for the schemes studied in [195].

	Estonian IVXV	NV-Civitas	KTV-Helios	BeleniosRF	Selene	Eos	Selections
ElGamal encryption	●	●	●	●	●	●	●
PKI	●	●[1]	●	●	●[1]	●[1]	○
Benaloh challenge	○	●	●	○[2]	○	○[3]	◍[12]
Threshold decryption	●	●	●	◑	●	●	●
Provable decryption	●	●	●	●	●	●	◍
Casting a re-vote	●	●	●	○	◑[4]	●	●
Proof of vote validity by voting application	○	●	○[13]	○	◍	◑	◍[12]
Cast-as-intended	●	●	●	○[5]	●	○[11]	◍[12]
Recorded-as-cast	●[6]	●[7]	●[8]	●	●	●	●
Tallied-as-recorded	○	●	●	●[9]	●	●	●
Mix-net	●	●	●	◑	●[10]	●	●
Homomorphic tally	○	○	○	●	○	○	○
Ring signatures	○	○	○	○	○	●	○

● = is used / supported ○ = not used ◑ = optional ◍ = unspecified

[1] PKI is not explicitly mentioned, but its functionality is implicitly described.
[2] Usage of Benaloh challenge is not mentioned in [65] and neither in the source code of the implementation: `https://github.com/webmaster128/private-voting/tree/master/packages/libbeleniosrf`.
[3] It is assumed that the voter has a trusted HSM for casting a vote [255].
[4] Whether re-voting is allowed in Selene depends on the used policy [280].
[5] It is assumed that voter's devices are honest [65].
[6] Individual verification can be used to check that voter's ballot has been received by the Vote Collector and registered by the Registration Service. However, individual verifiability does not detect maliciously cast re-votes [161].
[7] It is assumed that voters use smart card readers that contain a display and a PIN-pad [231].
[8] It is assumed in KTV-Helios that a malicious device can not use the secret key to re-vote [200].
[9] Holds only in case registrar and voting server are not maliciously colluding [65].
[10] Selene uses a re-encryption mix-net for assigning tracker numbers for plaintext votes.
[11] It is assumed that voters cast a vote via trusted HSMs that behave honestly [255].
[12] The format of the ballot and the procedure for creating it is unspecified in Selections [74].
[13] KTV-Helios can not use such proofs, but invalid votes are removed before tallying [200].

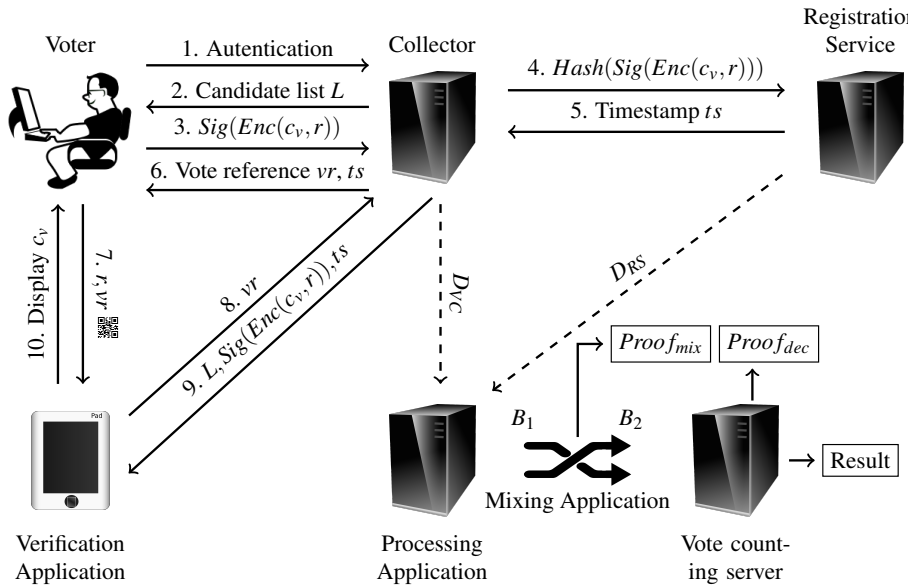


Figure 1. The figure gives an overview of the Estonian i-voting system, which has been used since 2017. The vote is denoted by c_v and randomness used for ElGamal encryption by r. The dashed arrows describe the processes that happen once the voting period has ended. In these processes, the collector and the registration service deliver the signed ballots (D_{VC}) and the signed timestamps (D_{RS}) to the processing application. More information about the depicted system can be found from [243, 169, 165]. The illustration is based on the figure from [33].

it did before the upgrade in 2017. The voter authenticates themself with a state-provided electronic identity (ID card or Mobile-ID), which relies on public key infrastructure. Access to the electronic identity is not an issue as the ID card is mandatory for Estonian citizens starting from the age of fifteen [3]. In addition, it is common in Estonia for people to use electronic identities to access e-services [290, 308].

After being successfully authenticated, a list of candidates is displayed to the voter. The voter can select one candidate. After the selection, the voting client encrypts the vote and asks the voter to use an electronic identity to digitally sign the encrypted ballot. Subsequently, the ballot is sent to the voting system, which returns a verification QR-code that is displayed by the voting application. The QR-code contains a vote reference and the randomness that was used for ElGamal encryption of the ballot. The former allows querying the corresponding ballot from the voting system, while the latter makes it possible to verify whether the vote was cast-as-intended and recorded-as-cast. The voter also has the option to re-vote to replace the previously cast vote. The amount of re-votes is unlimited.[4]

[4]During the parliamentary elections in 2011, one voter used the re-voting option more than 500

IVXV improves the verifiability of the voting system by tracking the ballots from the time they are submitted to the voting system until the encrypted votes are decrypted. Since the introduction of IVXV, trust has been divided between two independent organisations, one running a registration service and the other running the vote collector servers. The registration service keeps a private append-only bulletin board that registers all received ballots. Once the ballot is submitted, it is stored by a vote collector server along with the registration confirmation. During individual verification, voter's verification application also checks that the vote was registered with the registration service.

After the voting period ends, the information stored by the vote collector and the registration service is moved to the ballot box processor, which verifies the corresponding information, removes duplicate ballots, and checks whether ballots were cast by eligible voters. Next, signatures are removed from the encrypted ballots so that a mix-net can process them. The mix-net shuffles and re-encrypts the ballots and outputs a list of re-encrypted ballots along with a proof of correct shuffle. Finally, the set of anonymous encrypted ballots can be decrypted. After threshold decryption, the votes can be tallied. For the decryption, a proof of correct decryption is generated.

3.3.1. Discussion

While IVXV increases the verifiability of the server-side components, it does not provide universal verifiability, which would allow voters to check whether their votes were included in the tally.[5] There is no public bulletin board, and the voters do not have the information that is needed to verify the processes. The main reason for such a choice is to provide coercion-resistance for the voters. By the design of the voting scheme, a coercer should not have the means to check how a voter voted or whether a coerced voter cast a re-vote. The auditing and verification are made possible for auditors but only on the premises of the election organiser. Thus, the usage of independent auditors is critical for guaranteeing that the processes were handled according to the requirements.

Voters themselves are not able to check that only eligible voters participated in the elections. However, this is done by the vote collection service by querying different registries managed by the state (for example, the state has an electronic population registry). On the one hand, such a design protects voters' identities and thereby lowers the probability of an attacker forcing a voter to abstain. On the other hand, voters have to trust that the state handles the registries correctly. While there is no easily accessible public information regarding the identities of people who participated in the elections via i-voting, the election officials can monitor whether a certain voter abstains. The ballots submitted through the i-voting system contain signatures, which uniquely identify the participants. In

times [168].

[5] An overview of the verifiability properties and definitions can be found in [165].

addition, up to 2019, the voter lists in the polling stations contained information about the voters who cast their votes over the internet during the pre-election phase. This was necessary to avoid double votes as it was impossible to re-vote on paper during the election day. However, since 2021 it is possible to cast a paper-based re-vote also during the election day as the voter list is handled via a new information system [10, 7].

IVXV does not use zero-knowledge proofs to guarantee vote well-formedness. However, the official voting client only lets voters to cast valid votes. Still, by using an unofficial voting client, it is possible to cast an invalid vote. Thus, there is a theoretical possibility for an attacker to coerce voters into casting an invalid vote, but that would require the attacker to provide the voter with an unofficial voting client. As there is no public bulletin board, the invalid vote would not be published after being decrypted. In theory, the value of the non-existent candidate number could leak in the tallying phase or while studying anomalies. However, the spread of such information can be limited by procedural means. Thus, an attack that aims to force voters to cast invalid votes does not scale and requires insider access to be successful. Without having insider access, an attacker can not check whether the coercion succeeded as voters can overwrite the invalid vote by casting a re-vote.

To run the simulation attack, an attacker would have to access voter's credentials. According to the current knowledge, it is not possible to extract the private keys from the ID cards[6] or from the SIM card of the Mobile-ID. Thus, the coercer would have to physically approach the targets to capture their ID cards or Mobile-ID SIM cards along with the corresponding PIN codes. However, this alone is not sufficient to guarantee that the simulation attack succeeds as the victim can use an alternative ID to cast a paper ballot in a polling station, which overrules i-votes. Thus, the coercer would also have to confiscate the passport, driver's licence, and other identity documents, which could be used for voter identification in the polling stations.[7] Unless all of these are captured by the coercer, the coercer would have to monitor the voter during the whole voting period. While such an attack might work for a carefully selected group of targets, it does not scale well. In addition, it is questionable whether voters could be persuaded to share their electronic identities as these can also be used to access bank accounts and issue legally binding signatures.

It can be argued that IVXV does not have receipt-freeness as individual verification can be used to get a signed proof of the cast vote. Still, as the voter can re-vote, the coercer does not get a proof that the verified vote will be included in the tally. The coercer would have to confiscate all of the voter's identity doc-

[6] It was revealed in 2017 that about 750 000 Estonian ID cards were affected by the vulnerability in Infineon's RSA key generation algorithm [227, 25, 254]. The certificates corresponding to the vulnerable ID cards were suspended, and the card owners were given an opportunity to update their ID cards.

[7] This would be a criminal offence according to §162 of the Estonian Penal Code [5]

uments to ensure that the voter can not re-vote. However, violating freedom of election is a criminal offence according to §162 of the Estonian Penal Code [5].

In general, the only anti-coercion measure in IVXV is re-voting.[8] One could argue that the possibility to re-vote also provides over-the-shoulder coercion-resistance as the voter can allow the coercer to watch and influence how the initial vote is being cast. However, this only holds when the coercer cannot monitor the voter during the whole voting period. Before 2021, the weak spot in the design was the end of the i-voting period, which allowed the coercer to force a vote to be cast just before the end of the voting period. At that time, the only mitigation measure against such an attack was to visit the polling station after the i-voting period ended as the polling stations were open for two hours longer. Since 2021, it is possible for i-voters to cancel their i-vote also on the election day by visiting the polling station and casting a paper vote.

An overview of the assumptions used by IVXV and the achieved coercion-resistance properties can be seen in Table 2.

3.4. NV-Civitas and the JCJ/Civitas family

In the nineteen-nineties, receipt-freeness was considered an important property by researchers who studied voting schemes [51, 282, 246]. However, the other practical aspects relevant to coercion-resistance were often ignored. That changed in 2002 with the paper by Juels, Catalano, and Jakobsson [180]. They defined the concept of coercion-resistance, which included the issues of forcing a voter to abstain, casting an invalid vote, and getting access to voter's credentials. In addition, the paper [180] introduced the JCJ voting scheme, which was designed to be coercion-resistant while providing verifiability.

JCJ envisioned a voting process where voters are not explicitly identified before they cast a vote. Instead, the validity of ballots is later blindly checked against a list of voters. Coercion-resistance is achieved in JCJ by allowing voters to use fake credentials while they are being coerced. The real credentials can be used to re-vote later. The scheme relies on the assumption that the voter should have access to an anonymous channel during the voting process. The design of JCJ ensures that the coercer cannot check whether a voter has cast a ballot, which should effectively eliminate the threats of forced abstention and forced casting of an invalid vote.

JCJ was taken as a basis for the Civitas voting scheme, which was introduced in 2007 and published in 2008 [75]. Civitas extended JCJ in multiple aspects. It distributed the trusted voter registration authority between a registrar and a mutually distrusting set of registration tellers. Besides that, Civitas introduced ballot boxes in addition to the bulletin board and added support for approval and ranked voting. It also specified how credentials are distributed and how cryptographic

[8] The list of voters who participated in the elections is not published. This could be seen as a measure to prevent attackers from being able to check whether voters abstained.

components are used. Both JCJ and Civitas rely on mix-nets to remove identities from votes before tallying. It is stated in [75] that a real-world deployment of Civitas requires public-key infrastructure in order to bind keys to identities. Civitas provides universal verifiability by allowing everyone to verify the tabulation proofs. In addition, voters can check whether the tabulation tellers receive their votes.

In 2012 Neumann and Volkamer proposed an improved version of Civitas that was aimed to resolve some of its issues when deployed in practice [231]. We refer to this scheme as NV-Civitas. That paper describes the modifications to Civitas but omits the detailed description of the scheme and refers the reader to [287] for the details regarding Civitas. NV-Civitas is built upon the recognition that Civitas uses strong trust assumptions and has issues with usability. One of the main complexities of Civitas lies in the handling of fake credentials. It is assumed that voters can manage a set of fake credentials while preventing them from falling into the hands of the coercer. In addition, the voter has to be able to distinguish the genuine credentials from the fake ones to prevent casting an invalid vote by mistake. Neumann and Volkamer proposed to solve the credential handling in NV-Civitas by relying on smart card-based credentials. A brief overview of the scheme is given in Table 1.

3.4.1. Discussion

NV-Civitas is claimed to be both receipt-free and coercion-resistant. While it was designed to improve usability compared to Civitas, it is questionable whether the design can be implemented in practice. Table 2 highlights the main assumptions the scheme relies on, which form the basis in achieving the desired anti-coercion properties.

The cornerstone of the assumptions for NV-Civitas is the existence of voter-specific hardware, which consists of a smart card and a PIN-pad-based card reader that integrates a display. The latter is considered a trusted device that is not affected by malware. Unfortunately, such devices are not common. In addition, only some smart card readers provide a PIN-firewall that only allows the PIN to be inserted from the integrated PIN-pad. In addition, the specific requirements set on the software also imply that the smart cards featuring non-standard functionalities are not available on the market. Smart card vendors have revealed that they are reluctant to add non-standard functionalities to their smart cards [195]. The new smart card functionalities would require implementing, testing, and certification. However, the certification that is done according to Common Criteria or FIPS-140-3 is rather expensive and time-consuming. Thus, the financial burden associated with the new features might not be beneficial unless a substantial procurement is made. As an alternative, it might be possible to implement the non-standard functionality in-house. However, that would mean that the whole life-cycle of the smart cards and the corresponding software would have to be

supported locally. Doing that just for i-voting would probably make the cost of a single vote prohibitively expensive. Nevertheless, it was shown in [230] that from the technical point of view, it is possible to implement NV-Civitas on smart cards with tolerable performance.

In general, the idea of using special trusted hardware can solve many of the existing security issues related to i-voting. By extending this idea, it might be possible to create personal voting devices based on trusted hardware. However, unless the voters build the devices themselves, it is still necessary to trust the vendors. The same holds for the vendors of smart cards, but practice has shown that such trust can not be absolute. Numerous examples show that implementing the software for smart cards is error-prone [253]. In addition, vendors may violate the agreements that force private keys to be generated only in the chips of the smart cards [253].

When designing voting schemes, one of the goals is to reduce single points of failure that could result in corrupt officials or politicians being able to change the election result. Thus, it becomes questionable whether voters would trust special-purpose smart cards produced by the election organiser, especially if they would be used solely for elections.

An assumption that originates from Civitas is the usage of an anonymous channel to cast votes [287]. The paper introducing NV-Civitas proposes to use Tor as the anonymisation network [231]. While being rather easily usable, integrating Tor into a voting system is problematic for several reasons. First, as a large percentage of its traffic is related to illegal activities, it might prevent some people from using the voting software out of fear that such activities might be tied to them. Second, the illegal activities and the possibility to access uncensored information have caused multiple states to block access to Tor.[9] Such restrictions may also be used in corporate networks. As a result, it is difficult to estimate where Tor is blocked and bypassing the restrictions is not straightforward.[10] Thus, making a voting client dependent on access to the Tor network is a questionable decision as it might prevent some eligible voters from being able to carry out their right to vote. In addition, relying on Tor creates a single point of failure, which depends on an external factor that the election organiser does not control. This may lead to an attacker selectively blocking access to Tor in an attempt to influence the election outcome.

[9]The most well-known case is the Great Firewall of China being used to block Tor (https://blog.torproject.org/closer-look-great-firewall-china). As a recent example, it was reported by OONI that Tor was blocked in Tanzania on the eve of the general election day on the 27th of October 2020 (`https://ooni.org/post/2020-tanzania-blocks-social-media-tor-election-day/`).

[10]It is difficult to get precise information on the level of blocking happening in the Tor network. However, some information can be gained by monitoring anomalies, which hint at censorship (`https://metrics.torproject.org/userstats-censorship-events.html`). Another metric is based on the percentage of users who rely on bridges to access Tor (`https://metrics.torproject.org/userstats-bridge-table.html`).

NV-Civitas is receipt-free because the coercer cannot distinguish ballots cast with fake credentials from ballots cast with genuine credentials. The voter is displayed the hash of the ballot while casting a vote. While this gives the voter the ability to check whether the ballot was recorded-as-cast, it can not be used as a receipt as the bulletin board may contain both valid and invalid ballots cast by the same voter. The receipt-freeness depends on the supervised registration process being free of coercion, allowing the voter to privately pick a valid PIN code. By hiding the valid PIN code from the coercer, the scheme becomes over-the-shoulder coercion-resistant. Even when a coercer monitors the voter or asks the voter to record the vote casting process, the coercer can not distinguish the usage of a valid PIN from an invalid one. Thus, only the voter knows whether the cast vote will be counted in the tally. However, the downside of this approach is the system being fragile to typos, which can cause voters to accidentally cast an invalid vote. The same can happen with voters who forget or mix up PIN codes from different smart cards.

NV-Civitas is resistant to all of the three additional coercion factors described by Juels *et al.* in [181]. Simulation attack is not relevant as voter's credentials are protected by a smart card and a PIN code. In addition, providing a wrong PIN code to the coercer effectively blocks the simulation attack without the coercer being able to determine whether the PIN code is valid. However, it is unclear whether the implementation would allow the voter to change the valid PIN codes. That functionality would increase the usability of the system but also give the coercer additional information about a valid PIN code.

NV-Civitas does not allow invalid votes to be cast and tallied as the ballot contains a proof of well-formedness of the vote [230]. Thus, an attacker will not succeed in forcing the voter to abstain via casting an invalid vote.

It is impossible to observe the voting event via monitoring the network as votes are cast over Tor. In addition, the bulletin board does not contain signatures or other information that identifies the voters. Thus, if the coercer wants the voter to abstain, he would have to monitor the voter for the whole voting period. The only reliable way to force the voter to abstain is to confiscate the smart card required to cast a vote.

An overview of the assumptions used by NV-Civitas and the achieved coercion-resistance properties can be seen from Table 2.

3.5. Selections

Selections [74] is a voting scheme that Clark and Hengartner proposed in 2011. It belongs to the family of voting schemes that are built on the basis of JCJ [181]. Selections differs from JCJ and its derivates in the way how voters' credentials are created and used. It uses a system based on panic passwords [73], which allows the voter to simultaneously signal coercion to the election organisers and cast a fake vote that won't be counted in the tally. A brief overview of Selections is

given in Table 1.

Selections can be divided into six main sub-protocols: registration set-up, voter preparation, registration, election set-up, vote casting, and pre-tallying.

During the registration set-up, the trustees distributively generate private key shares and a public key for threshold ElGamal encryption. Voter preparation lets the voter create a set of passwords out of which the correct password and the panic passwords will be selected. These passwords will be encrypted with the public key of the trustees, and the resulting ciphertexts are printed on a piece of paper in a machine-readable format. The valid password will be selected during registration, which is done in-person in a private booth. To register, the voter has to be identified, which is a precondition for using the registration booth. The booth contains a computer that can read the previously created list of ciphertexts. The computer re-randomises the ciphertexts and prints the resulting values to a piece of paper along with the used randomness and initial ciphertext. Each row of the paper contains the new ciphertext and the information on how the ciphertext was created. However, the latter is printed on a scratch-off surface, which can be scratched to delete the randomness and the original ciphertext. The voter selects one ciphertext, which corresponds to the valid password that can be used to cast a valid vote and scratches off the information about the used randomness and original ciphertext. The corresponding re-randomised ciphertext is posted to a public bulletin board along with the VoterID.

The registered passwords should be usable across multiple elections. To set up new elections, the ciphertexts in the bulletin board are copied to a new election-specific bulletin board and blinded by the trustees.

It is important to note that vote casting is supposed to happen over an anonymous channel. The paper that introduced Selections proposed integrating an anonymous remailer or onion routing into the voting client [74]. To cast a vote, the voter has to submit a tuple consisting of a ballot, commitment to a password (or panic password), a re-randomised version of the value posted to the election-specific bulletin board, and two zero-knowledge proofs. The ballot structure is left unspecified with the requirement that it should be submittable to a mix-net. The first zero-knowledge proof shows that the re-randomised version of the election bulletin board value belongs to a set of values chosen from the same bulletin board. The second proof shows the knowledge of the committed password.

Finally, in the pre-tallying phase, the zero-knowledge proofs are checked, and invalid submissions are removed. In addition, duplicate votes are removed so that only the most recent remains. Next, the tuples are sent through the mix-net. The resulting list of anonymous tuples is used to check the validity of the used password. For each tuple, a plaintext equivalence test is run between the committed value of the password and the re-randomised value from the election bulletin board. Finally, the remaining valid ballots are decrypted by the trustees by using threshold decryption.

3.5.1. Discussion

Selections is claimed to be end-to-end verifiable and coercion-resistant against a non-adaptive adversary [74]. However, the paper introducing Selections does not specify which definition of end-to-end verifiability is followed [74]. In addition, it is not specified how a voter could verify that a malicious voting device does not replace the vote while the ballot is formed. When considering eligibility verifiability, the registration process is designed to identify voters, which would allow the authorities to check eligibility. However, it is unclear how the general public could verify that the election organisers do not cheat by adding non-existent voters to the bulletin board.

It is assumed that the voters are coerced after the registration phase [74]. While registration is done in a controlled environment, the booth does not prevent the voter from cooperating with the coercer. For example, there is an assumption that the voter has to erase the randomness used to re-randomise the selected password. However, it is easy to take a photo or make a video stream in the booth and thus leak which password was selected. Still, it is stated that the registration process has to be used only once, and it can be used to participate in multiple elections. The temporal aspect reduces the options for coercers to interfere with the registration. However, the same temporal aspect raises the question of storing the panic passwords. It is unlikely that the panic passwords could be memorised as they are rarely used. Thus, they have to be either written down or stored digitally. Consequently, the possibility of coercion would increase, especially if the coercer is a family member.

Another unspecified part of the scheme is the meaning of the VoterID. The entries in the public bulletin board contain the VoterID along with the re-randomised password. Depending on the specification of the VoterID, it may be possible to identify voters who do not abstain.

As the structure of the ballot and the processes regarding the handling of the ballot were not described, it is unknown whether voters could be forced to cast invalid votes. Thus, the possibility of being resistant to casting an invalid vote depends on the implementation, which means that different implementations of Selections might behave differently regarding this aspect.

However, the scheme provides resistance to a simulation attack as it allows a coerced voter to revoke their online voting registration and instead vote in person. While being introduced with good intent, the same feature could be abused by a coercer to force voters to abstain. While i-voting is supposed to allow remote voters to participate in elections, these voters may not have the opportunities to visit the polling stations. The knowledge of such a situation would increase the efficiency of the coercion attack.

As a whole, the scheme offers voters a way to evade coercion even in the presence of the coercer, but this comes with a price to usability. First of all, it is difficult for voters to understand how the scheme works without having a basic un-

derstanding of the underlying cryptographic building blocks. Second, the preparation, registration, and handling of the panic passwords is non-trivial. Third, it is assumed that voters can use an anonymous channel to cast a vote. However, there are very few options for implementing an anonymous channel in practice, with Tor being one of the possibilities. But using Tor for legally binding elections is questionable given the multiple issues described in Section 3.4. An overview of the assumptions used by Selections and the achieved coercion-resistance properties can be seen from Table 2.

3.6. Helios and KTV-Helios

Helios is a web-based open-audit voting system that Adida presented in 2008 [15]. It allows anyone to audit the integrity of the elections, but only voters themselves can check whether their vote was cast-as-intended and recorded-as-cast. The latter is implemented via the Benaloh challenge [47] and by allowing the voters to compare the hash value of their encrypted ballot to the one listed on the bulletin board. It is important to note that voters cannot cast a re-vote and that the randomness used to cast a vote can be seen as a receipt. The bulletin board contains the names of the voters along with their encrypted ballots. While such a design makes it easy for a voter to verify the vote, it does not guarantee on its own that all of the listed voters are eligible.

Helios is not suitable for elections that have a high risk of coercion as it is not receipt-free. The authors of Helios explicitly stated that Helios does not attempt to solve the coercion issue [15]. Nevertheless, in a low coercion setting, it provides an opportunity to run fully auditable elections. Helios achieves ballot secrecy by encrypting the votes and by using a mix-net before the tallying phase to separate identities from encrypted votes. Proofs of correctness show that mixing and decryption were done honestly. Third parties can check these proofs by downloading the full election data to verify that the mixing, decryption, and tally were done correctly.

There are multiple derivates and extensions to Helios. We focus on the two schemes, which aim to achieve receipt-freeness. The current section gives an overview of KTV-Helios [200] and Section 3.7 describes BeleniosRF [65].

KTV-Helios was introduced in 2015 by Kulyk, Teague, and Volkamer [200]. Compared to Helios, it adds private eligibility verifiability and receipt-freeness. The former allows anyone to check that only votes from eligible votes are included in the tally, thereby eliminating the risk of ballot box stuffing remaining undetected. Importantly, the method used for eligibility verifiability does not reveal whether the voters participated in the elections. This is achieved by publishing the list of eligible voters on the bulletin board and hiding the valid votes between dummy votes.

Dummy votes are the central feature of KTV-Helios, which makes it possible to hide re-votes from the coercer. KTV-Helios allows voters to cast a re-vote,

which contains a differential vote update. As multiple encrypted votes can be cast in the name of a single voter, the final vote is found by relying on the homomorphic property of the ElGamal encryption scheme.

After the election period ends, each voter's final encrypted vote is computed by performing element-wise multiplication of the ciphertexts cast for the given voter. As null votes are represented as encryptions of the value 1, they do not change the multiplication result.

While a voter is able to cast a differential vote update to change the previously cast vote, it should not be possible for an observer who monitors the bulletin board to distinguish null votes from valid ones. As it is unlikely for all voters to understand the system or bother to cast multiple null votes, this task is given to a separate party called posting proxy. Both valid and null votes are cast over an anonymous channel, which prevents a network-level observer from distinguishing null votes from legitimate ones. In addition, the posting proxies work in a randomised manner to avoid timing side-channels (see Section 3.8). The posting proxies also ensure that each eligible voter receives a sufficient number of null votes. This results in a receipt-free voting scheme as the voter can not prove that a differential vote update has not been cast. Still, it is questionable whether the voters would accept the fact that their vote is computed from multiple encrypted votes cast by other parties. Although the underlying cryptography requires access to the voter's private key to cast valid votes or differential vote updates, it may not be sufficient to gain the voters' trust.

The authors of KTV-Helios admit that the scheme is not coercion-resistant as it is possible to force voters to abstain or to cast an invalid vote [200]. While the latter is possible, it is difficult for an attacker to check whether the voter followed the instructions. KTV-Helios uses plaintext equality tests to remove invalid votes before they are tallied. In addition, if the voter knows the value of the invalid vote, it is possible to cast a re-vote for a suitable candidate. However, the voter does not have a counter-strategy if the coercion happens at the end of the voting period. An active attacker who can monitor the voter at the end of the voting period can force the voter to cast an invalid vote. The Estonian IVXV scheme faces a similar issue, but it co-exists with a paper-based voting system, which allows the voter to cast a paper vote that overrules the i-vote.

When considering the possibility of a simulation attack, KTV-Helios can be compared to the Estonian IVXV scheme. Both of these schemes rely on a PKI and store voters' private keys on smart cards. If the coercer forces the smart card to be used to vote in a specific way, there is the possibility to re-vote later to invalidate the vote given under coercion. Actually, KTV-Helios provides the voter with a counter-strategy, which prevents the coercer from successfully casting a vote even when getting access to the voter's smart card along with the PIN codes. Namely, the voter has to cast a vote at the beginning of the voting period without revealing the fact that a vote was cast. Thus, the coercer would have to confiscate the electronic identities before the voting period begins for a successful simulation

attack. While possible, such an attack does not scale well as it is likely that the coercer does not want to get caught. In addition, when the voting system provides an option to overwrite the i-vote with a paper vote, the coercer would also have to physically monitor the voter to prevent a paper vote from being cast.

3.7. BeleniosRF

BeleniosRF is a voting scheme proposed by Chaidos *et al.* in 2015 [65]. The scheme is built on the basis of Helios and aims to provide receipt-freeness while remaining end-to-end verifiable. While KTV-Helios relies on dummy votes to add receipt-freeness to Helios, BeleniosRF uses randomisable ciphertexts to achieve a similar goal. The aim is to prevent the voter from using the randomness as a receipt. Therefore, a randomisation service is introduced, which re-randomises the ballots before posting them to the bulletin board. BeleniosRF uses an unforgeable signature scheme that allows signatures to be adapted to re-randomised ciphertexts.

3.7.1. Discussion

Compared to the other voting schemes reviewed in this chapter, BeleniosRF relies on the least number of assumptions. It is stated that each voter requires a signature key pair, which hints at the need for a PKI. The scheme stands out by the way how the privacy of votes is achieved in the tallying phase. Namely, BeleniosRF allows the election organisers to choose whether to use homomorphic tallying or rely on a mix-net to separate voters' identities from ciphertexts.

While BeleniosRF enables receipt-free voting, it does not achieve coercion-resistance. Compared to other schemes, the most significant outlier in the context of coercion-resistance is the impossibility of casting a re-vote. The authors of BeleniosRF argue that this is not a significant issue as re-voting is not common in practice and is not supported by the legislation in most countries. Nevertheless, the scheme also does not use alternative mitigation measures like the possibility to cast fake votes, which means that it does not protect the voter against a coercer who is present while the vote is being cast.

Although the scheme incorporates cryptography to achieve receipt-freeness, that may not be sufficient in practice. The easiest way for a voter to create a receipt is to video record the vote casting process. As BeleniosRF does not allow to cast re-votes, the voter can not invalidate a vote cast under coercion. As a counterargument, one could claim that video recording is not reliable proof as it can be done using a mock-up of the voting software. However, it is not realistic to assume that an average voter can fake the vote casting procedure. At the same time, if a coercer aims to change the outcome of the elections, it does not matter if a small percentage of the victims can outsmart the coercer.

BeleniosRF does not protect against the three coercion methods described by Juels *et al.* in [181]. It is possible to force the voter to abstain or to share the

voting credentials. In addition, no specific countermeasures were described to prevent voters from casting invalid votes. An overview of the issues that enable such coercive behaviour can be found in the next paragraphs.

The bulletin board contains re-randomised signed ballots and the corresponding public keys of the voters who cast the votes. Thus, by assuming that the public keys can be tied to voter identities, it is possible to check which voters participated in the elections. If there is no public database to link public keys to identities, an adversary could request the public key from the coerced voter. Therefore, BeleniosRF does not provide protection against forced abstention attacks.

BeleniosRF uses homomorphic tallying but does not specify whether there are any restrictions on casting or tallying invalid votes. While the message space for encoding the candidates is limited, it may be possible to encode an invalid vote. Thus, the resistance to the coercer forcing the voter to cast an invalid vote depends on the scheme's implementation.

The scheme does not specify whether special hardware should be used to manage voter's signing keys. In case the keys are stored in the computer's memory, it is possible to copy and share them with a coercer. Even when the scheme would integrate hardware-based key management, it would not prevent the voter from selling the vote by sharing the hardware token with the coercer.

The only effective counter-strategy against an active coercer is to cast a vote at the beginning of the voting period, thereby making it impossible for the coercer to use the voting credentials. Importantly, the strategy does not work when the coercer tells the victim that it is possible to check whether the voter has already cast a vote. Thus, when the coercer confronts the victims before the voting period, they either have to ignore the threats (and possibly report them) or allow the coercer to use their credentials. However, ignoring the threats can have negative consequences for the victim. As a result of the previously stated issues, BeleniosRF does not protect against a simulation attack.

3.8. Selene

Selene is a voting scheme that was introduced by Ryan, Rønne, and Iovino in 2015 [279, 280]. The main distinguishing factor of the scheme is that the voters do not have to deal with excessive cryptography. Once votes are decrypted, they are posted to the bulletin board along with tracking numbers that can be used for vote verification. To prevent the tracking numbers used as a receipts, the voters will be told which tracking number belongs to them only after the votes have been decrypted and tracking numbers published on a bulletin board. That design allows voters to present wrong tracking numbers to the coercers.

A valid signing key gives the voter eligibility to participate in the elections. While it is assumed that each voter has a key pair, it is not specified how the private keys are handled. When voters register for elections, tracking numbers are committed to a bulletin board by using Pedersen-style trapdoor commitments.

These commitments are tied to voters' public keys.

Ballot stuffing is avoided by requesting the encrypted ballots to be signed by the voters. Once the voting period ends, the signed and encrypted ballots are posted to the bulletin board next to the previously submitted commitments. Next, the identifiable information is separated from the encrypted ballots, which are sent through a mix-net. After the mixing phase, a threshold set of tellers decrypt the ballots and provide a decryption proof. As a result, pairs of tracking numbers and votes are published. After a while, the voter is sent the de-commitment value α to open the commitment and reveal the tracking number assigned during the registration phase. The tracking number can be used to verify that the voter's vote was decrypted and thus included in the tally. In case of coercion, the voter can create an alternative de-commitment value that is cryptographically indistinguishable from the real one.

The scheme is designed to provide receipt-freeness while offering both individual and universal verifiability. However, when considering coercion-resistance, many of the details are left unspecified. This is probably done on purpose as the authors state that Selene could be used as an add-on to other schemes. Still, the vagueness makes it difficult to analyse the coercion-resistance of the scheme. For example, the vote casting procedure is not specified, and the choice to allow re-voting is left open.

3.8.1. Discussion

Although the paper's title introducing Selene mentions coercion mitigation ("Selene: Voting with Transparent Verifiability and Coercion-Mitigation"), the authors state that the scheme is designed for low-coercion environments [280].

While the scheme is designed to be receipt-free, this can not be guaranteed if re-voting is disabled. Similarly to BeleniosRF, the voter could make a video recording of the voting process to sell the vote.

As the primary anti-coercion measure, the voter is given the possibility to fool the coercer by producing fake de-commitment values. However, that may not be sufficient as the coercer can request information regarding the way how the α was delivered. Thereby, the delivery information could also be used as a receipt. For example, it is important that the coercer would not be able to monitor the communication channels used to deliver the actual α term to the voter. Thus, the authors of Selene propose that the α term should be delivered over an unauthenticated and private channel. However, it is difficult to create such a channel in practice.

One way to prevent the coercer from monitoring the traffic is to rely on Tor. Unfortunately, this may not provide the required level of reliability as described in Section 3.4. Another option would be to use email as the distribution channel – however, the official email addresses and the corresponding metadata like DKIM signatures [102] function as receipts. Thus, the fake de-commitment values would also have to be sent from the same email address to convince the coercer. While

such problems could be solved by using random email addresses, that may create issues with usability and trustworthiness. The same also holds for other delivery channels.

Even when the delivery channel would not be an issue, the coercer should not be able to predict when the α is sent to the voters. Thus, the authors of Selene suggest that the delivery time should be randomised. However, that would lower the system's usability and might end up reducing the number of voters who verify their votes.

When leaving aside the problems with delivering the α value, the scheme has issues with the additional coercion methods described by Juels *et al.* [181].

The basic scheme presented in [280] does not hide the participants' identities as voters' signatures are posted to a public bulletin board. Thus, the basic scheme does not protect voters against forced abstention. However, an extended version of Selene is described in the eprint version of the paper [279]. The extended version contains an enhancement, which allows using pseudonymous credentials that hide voters' identities. However, it is admitted that the countermeasure provided by pseudonymous credentials could be bypassed in case the voter is forced to reveal the signing key. The authors of Selene proposed to solve the issue by using malleable signatures similarly to BeleniosRF [279].

Whether Selene prevents invalid votes from being cast depends on the implementation. Selene can be used as an add-on together with another voting scheme that prevents invalid votes from being revealed. For example, this property is present in a scheme proposed in 2017 that combines JCJ with Selene [175].

The description of the scheme does not mention any countermeasures against a simulation attack. The paper introducing Selene does not specify how voter's credentials are handled [280]. Unless the signing keys are stored in a hardware token in a copy-protected manner, it is possible to share the keys with the coercer. In addition, by enabling re-voting, it becomes more difficult for the coercer to guarantee that the simulation attack succeeds.

3.9. Eos

Eos is a voting scheme proposed by Patachi and Schürmann in 2017 [255]. The scheme relies on conditionally linkable ring signatures to hide the identities of voters. The ring signature allows voters to cast a vote that is signed anonymously in the name of the ring. The scheme allows voters to select between a valid identity and pseudo-identities, which can be used in case of coercion. Eos is designed to provide universal verifiability and is claimed to be coercion-resistant.

The authors of Eos propose using the hardware wallets designed to store cryptocurrencies' keys and repurpose them as personal HSM-s. The HSM would be a trusted device responsible for storing the voter's private key, authenticating the voter, encrypting the ballot, generating the ring signature, and handling the randomness used in these processes.

Eos is designed to allow each voter to have multiple pseudo-identities and one electoral identity. The scheme integrates the HSM to play a central role when choosing which identities to use. The input to the authentication method used by the HSM determines whether a valid identity is used to cast a vote. In case PIN codes are used for authentication, a voter has one valid PIN code, which can be used to force the HSM to cast a vote with a valid identity. When any other PIN code is used, a different pseudo-identity is used along with an invalid electoral identity, which results in the coerced ballot being discarded.

The scheme allows voters to signal coercion to the election organisers. This is implemented by the way how the ballot is formed. The ballot consists of three ElGamal tuples. The first tuple contains either an encryption of an election-specific generator or an encryption of 1. The paper refers to this tuple as the encryption of envelope colour [255]. The term "green colour" is used when the voter entered a correct PIN, in which case the election-specific generator is encrypted. However, if the voter entered a wrong PIN, an encryption of one is placed into the first ElGamal tuple, which is denoted as the red envelope. The second ElGamal tuple contains an encryption of voter's electoral identity or an encryption of one. Similarly, depending on whether the correct PIN code is entered, either a valid electoral identity or the value one is encrypted. The third ElGamal tuple contains the encrypted vote.

Eos uses three bulletin boards and two mixing phases to move the information between bulletin boards. The first bulletin board contains information that the voters sent to the voting system. For each interaction, it contains an encrypted ballot, a ring signature and a pseudo-identity assigned by the HSM. Once the voting period ends, the ballots with invalid proofs are discarded along with duplicate ballots. Next, the encrypted ballots are sent through a mix-net, after which the first two tuples of the ballots are decrypted by relying on a threshold number of tellers. The result is posted to a second bulletin board containing the ballot colors, electoral identities and encrypted votes. Ballots that contain either invalid electoral identities or that are referred to have red colour are discarded. The resulting encrypted votes are sent through a mix-net and decrypted by a threshold number of tellers. The outcome is posted to a third bulletin board, which now contains the decrypted votes. The latter can be used for a public tally.

3.9.1. Discussion

Eos provides mitigation against multiple types of coercion. The usage of mix-nets prevents a coercer or vote buyer from following the vote on the bulletin board. Thus, even the voter is not able to prove how the vote was cast. Suppose the voter has to confront a coercer who is physically present. In that case, it is possible to use a pseudo-identity, which highlights coercion and results in the coerced ballot being discarded. Once the voter can vote freely, it is possible to cast a re-vote with the correct identity, which results in the vote being tallied. Depending on the

implementation, Eos can also protect the voter from being forced to abstain, from being forced to cast an invalid vote, and from falling victim to a simulation attack. These aspects are covered in more detail in the following paragraphs.

Eos protects voters against forced abstention. The bulletin boards do not contain information about the voters' identities, and the votes are cast over an anonymous channel. Conditionally linkable ring signatures guarantee that the voter's identity is not linked to the signature unless it is the voter's wish. Even when a voter wishes to claim a signature to sell the vote, the mix-nets prevent the voter's identity from being linked to the vote.

Whether Eos prevents voters from being forced to cast an invalid vote depends on the implementation. The authors admit that the basic scheme would be vulnerable to such an attack and propose a mitigation. Namely, the voting protocol could force the voting software to provide a disjunctive zero-knowledge proof to guarantee that the cast vote belongs to a set of valid votes. Based on the validity of the proofs, only valid votes would be decrypted.

The probability of a simulation attack is low due to the credentials being protected by an HSM. However, if the coercer has physical access, the configuration of the HSM starts to play a role. The authors mentioned that, in theory, it would be possible to try out all possible PIN codes to cast a valid vote. They also pointed out that it is easy to mitigate such an attack by increasing the complexity of the PIN (or alternative authentication method). However, a more interesting question is related to the management interface of the HSM. Namely, what will happen if the real PIN leaks? If the HSM provides a method for changing the valid PIN, the coercer could also use it to identify a valid PIN. On the other hand, restricting the possibility to replace PIN codes would hurt usability and increase the cost of running the voting system. Even when a coercer would eventually be able to use the HSM, it would require significant effort. Thus, a simulation attack is unlikely to affect many voters.

3.10. General issues with achieving coercion-resistance

The meaning of privacy and the threat of coercion is not uniform globally due to the differences in cultures, societies, and how the countries are governed. Therefore, the voting systems have to be tailored to suit the specific context.

The varying meanings of privacy are illustrated by the perceptions of privacy between the western and eastern cultures. For example, in Japan, privacy is mostly community-oriented, while in western cultures, individual privacy is paramount [63]. However, the expectations for privacy differ also among the Western cultures, with Germany being an example of having strict boundaries and privacy regulations [260].

The historical memories of repressions based on ethnic or social classification can highlight the necessity to protect ballot secrecy. It is not inconceivable that in the future, some people could face discrimination based on their historic political

preferences. Thus, in some contexts, i-voting may only be considered if strong privacy guarantees can be provided.

The risk of vote-buying is directly related to the welfare of the society and the level of law-abiding citizens. Thus, rich and old democracies may not consider vote-buying as a significant risk, which is the position taken by Switzerland. For example, the specification document for the Swiss Post Voting System describes requirements for verification, but its threat model omits coercion and vote buying [265, 264].

However, when coercion is considered a threat, it is not straightforward to implement mitigation measures, as seen from the previous sections. It turns out that the common techniques used for providing ballot secrecy are not sufficient to prevent vote-buying and coercion. For example, i-voting schemes studied in [195] and described in this thesis protect voters' privacy in the tallying phase by either using mix-nets or homomorphic encryption. All of these schemes protect ballot secrecy by relying on encryption. While these approaches aim to prevent voters' choices from being leaked, they do not prevent the three types of coercive attacks described by Juels *et al.* [181]. Thus, voting schemes that aim at achieving coercion-resistance must include additional mitigation measures against these attacks. Table 2 summarises the main assumptions and coercion-resistance properties of the voting schemes reviewed in [195]. The selected coercion-resistance properties correspond to the requirements described in Section 3.1.

It was already mentioned in Section 3.1 that receipt-freeness does not guarantee coercion-resistance. In general, protecting a voter against an active coercer who is physically present is a difficult task that requires the cooperation of the voter. Unless the voter is aware of and able to convincingly apply the mitigation measures like fake credentials or re-voting, the coercer could force the voter to either abstain, cast a randomised vote, or acquire the voter's credentials to run a simulation attack. Deniable encryption [62] could be seen as a partial mitigation measure as it provides the voter with an option to lie to the coercer about the cast vote. However, it would not prevent the creation of a receipt if the voter is willing to cooperate with the coercer while the vote is being cast [172, 182]. It has to be understood that executing some of the aforementioned attacks is made easier if an attacker can monitor the bulletin board [182]. This once again highlights the trade-off between transparency and coercion-resistance.

Applying the anti-coercion mechanisms becomes more complicated if the coercer can remotely monitor the computer used to cast a vote. Such monitoring could be done via malware or side-channels. For the sake of simplicity, we can assume that the coercer has full control over the computer. An attacker with such capabilities could identify whether re-votes are used to cancel the votes given under coercion. Unless code-voting is used, such an attack could also identify which candidate the voter voted for. Thus, there are no simple solutions for the issues related to untrustworthy end-user devices.

One approach would be to cast re-votes from a different device, but voters

Table 2. Assumptions and coercion-resistance properties of the schemes that were reviewed in [195].

	Estonia	NV-Civitas	KTV-Helios	BeleniosRF	Selene	Eos	Selections
Special client hardware	●[1]	●	●	○	○	●	○
Anonymous channels	○	●	●	○	●	●	●
Fake credentials	○	●	○	○	○	●	●
Casting a re-vote	●	●	●	○	◑[2]	●	●
Non-trivial registration	○	◑[3]	○	○	○	○	●
Receipt-freeness	○	●	●	●	◐[4]	●	◐[5]
Over-the-shoulder coercion-resistance	●	●	◐[6]	○	◐[7]	●	●
Resistance to forced abstention	◐[8]	●	◐[9]	○	◑[10]	●	◐[11]
Resistance to casting an invalid vote	◐[8]	●	◐[12]	◑[13]	◑[14]	◑[15]	◑[16]
Resistance to simulation attack	◐[17]	●	◐[17]	○	○[18]	◑[19]	●[20]

● = is assumed / holds ○ = is not assumed / does not hold
◑ = depends on the implementation ◐ = may hold

[1] Smart card-based ID cards are mandatory in Estonia and widely in use.
[2] Whether re-voting is allowed in Selene depends on the used policy [280].
[3] Information about the registration process of NV-Civitas can be found in [231].
[4] Selene's receipt-freeness depends on the anonymous channel, see Section 3.8.
[5] Depends on handling re-randomisation randomness during registration, see Section 3.5.
[6] The property depends on how the coercer prevents re-voting, see Section 3.6.
[7] The property depends on the re-voting policy in the implementation of Selene [280].
[8] The attack can be implemented by an insider, see Section 3.3.
[9] KTV-Helios is susceptible to forced abstention only in the case of an active attacker.
[10] For information about the implementation of Selene, see Section 3.8.
[11] It is not clear whether Selections is resistant to forced abstention, see Section 3.5.
[12] Invalid votes can be cast but they are filtered out before tallying, see Section 3.6.
[13] See Section 3.7 for information about the coercion properties of BeleniosRF.
[14] Vote casting procedure is not specified in Selene, see Section 3.8 for more details.
[15] Whether an invalid vote can be cast depends on the version of Eos, see Section 3.9.
[16] It is not specified how the vote is encoded and how votes are tallied in Selections [74].
[17] The coercer might access the smart card and PIN codes. However, the voter can re-vote.
[18] It is not specified how keys are managed in Selene [280], see Section 3.8 for more details.
[19] Depends on the configuration of the HSM. For more information, see Section 3.9.
[20] It is possible to revoke the registration and vote in person, see Section 3.5.

may not have access to multiple devices. At the same time, using someone else's computer to cast a vote raises similar issues with privacy. Another possibility is to use special trusted voting devices, which are unlikely to be controlled by an attacker. This kind of a solution was suggested by the authors of Eos [255], who proposed that a special HSM should be used for voting. Another recent example comes from our paper published in 2021, which describes how a proof-of-concept i-voting device was developed for the Estonian i-voting system [120]. However, even when it would be possible to design a secure, transparent and trustworthy personal voting device, doing this in practice would be expensive. For example, it is not sufficient to design a voting device, as also the supply chain would have to be secured to guarantee that the voters would get unaltered devices.

The latter brings us to non-trivial assumptions, which are sometimes made by the authors of the voting schemes [195, 199]. When non-trivial assumptions are made regarding the anti-coercion measures, it becomes questionable whether such mitigation measures can be applied in practice.

Besides the protocol-level issues, one of the main aspects to consider is the usability of the anti-coercion measures. While the voter does not necessarily have to understand the passive privacy measures automatically provided by the voting system, the knowledge and understanding of the provided anti-coercion measures is crucial for them to succeed. Unfortunately, there is a lack of usability studies that focus on coercion-resistance techniques.

The usability aspects of anti-coercion measures were covered in a paper published in 2020 by Kulyk and Neumann. The paper was based on a literature review that listed the commonly used assumptions and identified possible usability issues [199]. However, user experiments were not part of the research. As a rare example of practical usability research in the context of coercion-resistance, the usability of fake credentials was studied in 2018 by Neto *et al.* [228]. The experiments were based on the CIVIS system [228], which implements the voting protocol proposed by Araújo *et al.* [30]. Although the number of test subjects was rather small (80 in total), 30 participants out of 34, who took part in the third stage of the study, did not fully understand the purpose of casting fake votes. The study also revealed significant issues with the participants being unable to detect typos in passwords and thus not being able to distinguish whether a real or a fake vote was cast. Thus, the study raises the question of whether fake credentials can be applied at all in large scale elections. Therefore, the usability aspects of anti-coercion schemes require further research and prioritisation.

4. SIDE-CHANNELS IN VOTING ENVIRONMENTS

Security of a computer system has to be viewed as a whole by considering all of the possible attack vectors. Thus, besides malware and cyber attacks, other aspects like human errors, faulty procedures, and side-channels must be acknowledged.

There is a long history of side-channel attacks that goes back to World War II. In 1943, it was discovered that a teletype terminal 131-B2 used for encryption by the US army leaked the plaintext input due to electromagnetic emanation that could be intercepted from 80 feet away [131]. During the Cold War, it became a standard practice to use side-channels in espionage, which created a necessity to protect classified information against such interception.

While many side-channel attacks exploit the leakage of electromagnetic emanations, there are also other less obvious side-channels. For example, information can leak due to cache attacks, timing attacks, power consumption analysis, sound measurements, and detecting vibrations in different materials. In principle, any indirect event or emitted signal could be used as a side-channel. However, in practice, it may not be straightforward to measure the signals.

Elections and voting systems are no different in this context as they can also contain side-channel leakages. A side-channel attack against a voting system could result in ballot secrecy being violated or cryptographic keys being leaked. The most obvious sources for such leakages are computers that are used in election systems. These include the electronic voting machines and servers that tally the results. One well-known example of a voting machine-based side-channel originates from the Netherlands, where it was common to use electronic voting machines during elections. But that changed after 2006, once it was shown that the Nedap ES3B voting machines contained a side-channel that could leak the party preference of a voter [142, 261].

Although there is a lack of documented examples of side-channels being found in election systems, this does not mean that the problem is non-existent. First, the requirements set for the devices used in the election systems often do not consider side-channels [130]. Second, it is difficult for the researchers to access election equipment to examine the devices. Third, the possible existence of side-channels depends on the combination of hardware and software, which can change from election to election. Fourth, faulty implementations of cryptographic algorithms have been a cause of side-channel leakages [207, 19]. Fifth, prior research has shown that the common components used in computers often contain side-channels.[1] For example, such leakages have been found from keyboards [325, 52, 317], displays [198, 285], cables [289, 135], and printers [39].

In principle, it is possible to design election systems that are not vulnerable to side-channel leakages, but the process itself would be non-trivial and would have to be based on strict requirements. A paper published in 2011 by Frankland *et*

[1] Consumer devices are usually not designed to prevent side-channel leakages.

al. addressed this issue and defined requirements for voting machines that aim to prevent side-channel leakages [130].

Although it is possible to limit such leakages in a controlled environment, this may not be viable for end-user devices, which are used as voting clients in remote online voting systems. More surprisingly, it turns out that even the regular paper-based voting has acoustic side-channels, which could be used to violate the vote privacy provided by the voting booth [196, 197]. The acoustic side-channels against paper voting will be the main focus of this chapter.

4.1. Side-channels in paper voting

There are only a few options for side-channels leakages in regular paper-based voting systems when assuming that the attacker cannot modify or analyse individual ballot sheets. If other types of attacks are also considered, one could imagine lifting voters' fingerprints from the filled-in ballots to violate vote privacy. However, there is an easier way for an attacker to violate vote privacy. Namely, the content of the vote can be captured while the ballot is being filled in the voting booth.

The simplest way to do that involves an attacker placing a camera into the voting booth. However, it is questionable whether this could be done without the camera being detected, but, in some cases, that may not even be necessary. A voter can be coerced to use a smartphone to record the vote to prove to the coercer how it was cast.

Interestingly, vote privacy can be violated by a photo camera even when it is not possible to record the vote casting process. It turns out that photo cameras can capture the internal structure of the paper sheet, which can be seen as a fingerprint. The idea to fingerprint blank ballot sheets and find matching fingerprints from the filled-in ballots was described by Calandrino *et al.* in 2009 [60]. The attack assumes that it is possible to distribute ballot papers to voters in a manner that allows an attacker to link the fingerprinted ballot sheets with voters' identities and thereby violate vote privacy. The paper fingerprinting techniques were improved in 2017 by Toreini *et al.* who showed that the attack does not require expensive equipment [307]. The paper described that regular devices like an overhead projector and a photo camera could be used to fingerprint paper sheets. This is yet another example of attacks only getting better over time. In addition, it shows that the advancement of technology also has to be considered in the context of paper voting.

However, another approach for capturing the information on the ballot sheet involves intercepting the sound of the ballot sheet being filled in. In paper voting, the voters are expected to take a ballot sheet to the voting booth. Usually, the voting booth contains a pen and a table that allows the voter to either mark the selected candidates or fill in the write-in forms on the ballot sheet. While the booth and the curtain are meant to protect the voter, they also provide coverage

that may allow microphones to be hidden. It turns out that the sound of filling in the ballot sheets both emits sound and causes the table plate to vibrate slightly. Therefore, it is possible to attach microphones under the table to measure the sound of voters filling the ballots.

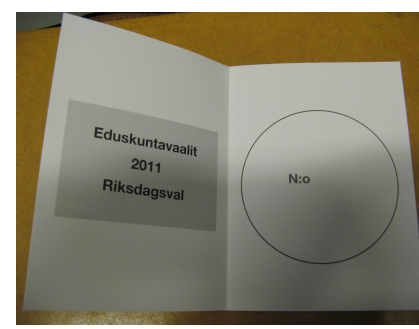
We hypothesised that it might be possible to use the acoustic side-channel to identify which candidates the voters are voting for. To check whether the hypothesis holds, a series of experiments had to be performed. However, the approach had to be tuned for the specific ballot sheet design as there are many different ways how voters can show their preference. For example, in some cases, the voters have to mark the checkboxes or preferences next to the candidates, while in other cases, the voters have to write either the name or the number of the candidate on the ballot sheet. To verify whether such a side-channel attack would be feasible, we initially limited the scope of our study to the ballots requiring voters to fill in the candidate numbers [196]. In the follow-up experiments, we showed that, in principle, it may also be possible to reveal the vote when the voters only have to mark the checkboxes next to the candidates [197].

4.1.1. Types of paper ballots

Before describing the experiments that tested the feasibility of using the acoustic side-channel, we give an overview of the common ballot designs used in different election systems.

In some election systems, the voters only have to cast a vote for a single candidate. For example, in Estonian and Finland, each candidate is assigned a unique number, and the voters are asked to fill in their ballot sheets with the number of the selected candidate. These ballots can be seen in Figure 2.

(a) Ballot paper used in Estonia for the municipal council elections in 2017 [242].



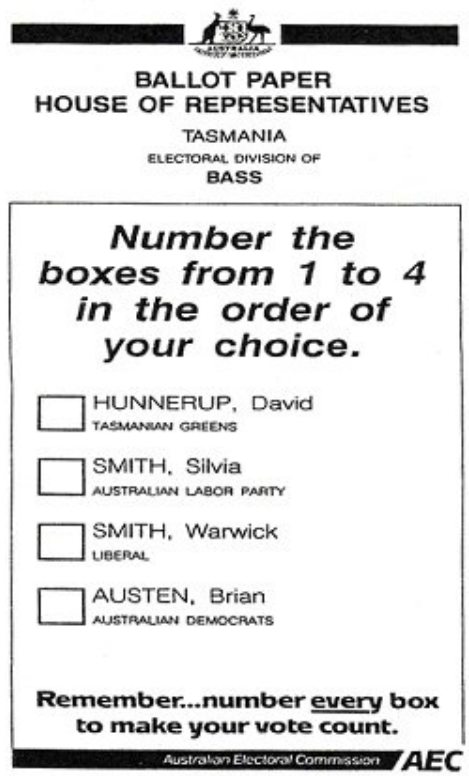
(b) Ballot used in Finland for the parliamentary elections in 2011. The same ballot design was also used for the 2015 elections.

Figure 2. Examples of ballots that are designed to be filled with candidate numbers.

The other types of ballots that often require voters to fill in numbers are used in election systems that rely on ranked voting. There are multiple preferential voting systems, but the ones commonly used for elections are single transferable vote (STV) and instant-runoff voting (IRV), which is also known as the alternative vote

(AV) [306]. Among other countries, STV is used in Australia, Ireland, Scotland, Malta, and Northern Ireland, while IRV is used for some elections in Australia, Ireland, Canada, and United States. An example of a ballot paper used for ranked voting can be seen in Figure 3. However, in some elections, the ordering of the candidates on the ballot paper is randomised to prevent giving some candidates an unfair advantage. This is relevant in countries where it is mandatory to participate in elections as it can cause some voters to rank the candidates in descending order.

(a) A sample of a filled-in ballot paper for electing members to the Australian House of Representatives. Image source: Parliament of Australia website, `https://www.aec.gov.au/Voting/How_to_Vote/Voting_HOR.htm`.



(b) Example of a ballot paper used for electing members to the House of Representatives. Image source: Parliament of Australia website. Australian electoral systems, `https://www.aph.gov.au/About_Parliament/Parliamentary_Departments/Parliamentary_Library/pubs/rp/RP0708/08rp05`.

Figure 3. In Australia, it is common to use ranked voting, which allows voters to divide the vote between multiple candidates. In such cases, voters have to write numbers to the ballot paper to mark the ordering of candidates.

In the context of acoustic side-channels, there are also other interesting ballot designs, for example, the ones that require voters to use textual write-ins. Such ballots are sometimes used in referendums and elections that allow the voters to

write the candidate's name on the ballot sheet. An example of a referendum ballot, which asks the voters to write either Yes or No, can be seen in Figure 4.

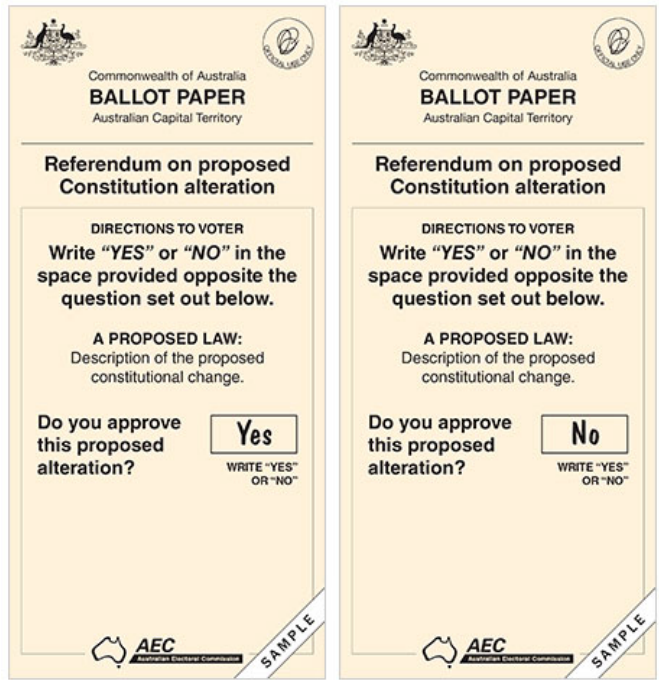


Figure 4. An example of an Australian referendum ballot. Image source: Parliament of Australia website, `https://www.aec.gov.au/Voting/How_to_Vote/voting-referendum.htm`.

Besides the already mentioned ballot types, many voting systems rely on ballots where the voters have to mark checkboxes to show their preferences. These ballots leak less information than those where the voters have to write numerical or textual information. However, the voter filling in the ballot can still leak information through vibrations on the surface where the ballot paper is being filled. It turns out that such leakages may be sufficient to figure out the selections on the ballot paper. The reason is that many of the ballots that contain checkboxes require a very large area to fit all of the candidates, and thereby the ballot is likely to fill a significant amount of the voting booth table. An example of a large checkbox-based ballot sheet can be seen in Figure 5. If the ballot sheet covers most of the table plate, it may be possible to use the audio channel to locate the signal source and identify the more and less likely selections. This could be done by relying on multiple microphones placed under the table to measure the signal's time difference of arrival (TDOA) [292, 151].

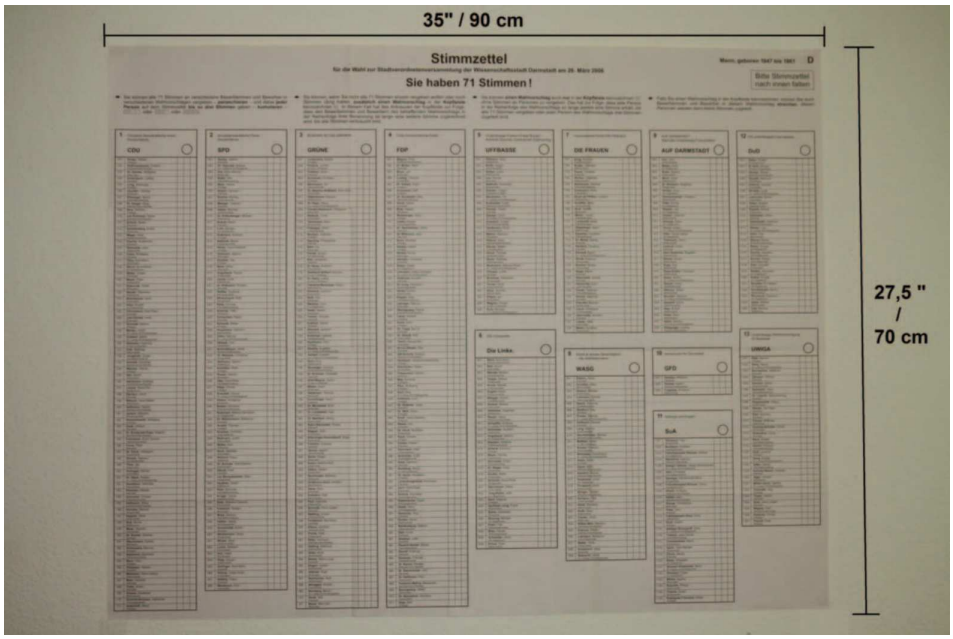


Figure 5. German local election ballot sheet [315], ©2011 IEEE.

4.2. A side-channel against paper voting – number detection

The idea of using microphones to reconstruct numbers is not new on its own. Researchers have shown that it is possible to identify the PIN codes based on the sound of buttons being pressed on the PIN-pad [106]. Similarly, it has been demonstrated that smartphone's microphones can be used to recover the typed in PIN codes due to the tapping events emitting sound [288]. However, there are also more exotic results, like the possibility to recover the text that is being printed out by listening to the sound of a dot-matrix printer [39].

Our study aimed to determine whether it would be possible to identify numbers based on the sound emitted when a pen is used to write numbers on a sheet of paper. To check the feasibility of such an approach, we used a readily available conference speakerphone Jabra Speak 410, containing a decent microphone. We placed the device on top of the table along with a piece of paper and started recording the audio emitted from writing numbers. Playing the recording proved that, in principle, such a side-channel may be possible as the sound of writing was clearly identifiable from the recording. An example of the recorded waveform can be seen in Figure 6.

4.2.1. Description of the data collection

Based on the initial results, we moved to the next phase of our study, which was to automate the number detection. As the audio recordings contained a pattern for

Figure 6. Waveform representations of numbers zero to four.

each number that was detectable by a human ear, it was likely that an algorithm could recognise these patterns. Thus, we continued the work to find a classifier that could label the audio samples with the corresponding numbers.

As a first step, we had to collect data, which could be used to test and train the classifiers. This meant that the configuration for the data collection had to be specified so that the recordings would be done under similar conditions. To compare the samples, the recordings would have to be captured with the same microphone and by having similar levels of background noise.

Thus, we had to select a suitable microphone for collecting the audio samples. We tested four different recording devices, and surprisingly the initially used Jabra Speak 410 did the best when capturing the emitted signal. We expected the semi-professional Rode VideoMic Pro to outperform Jabra Speak 410, but this was not the case. The reason may lie in Jabra Speak 410 having built-in digital signal processing (DSP), which removes noise [34]. Regardless, all of the tested microphones could only pick up a sufficiently good signal from a relatively close range. An overview of the tested devices can be seen from Table 3.

Table 3. Comparison of the tested recording devices. No technical specification was available for the microphone on the HP laptop and for the iPhone SE microphone. This table originates from [196].

Recording device	**Number of microphones**	**Type**	**Range**	**Sensitivity**
HP laptop	2	omni-directional	N/A	N/A
iPhone SE	3	omni-directional	N/A	N/A
Jabra Speak 410	1	omni-directional	100 Hz - 10 kHz	N/A
Rode VideoMic Pro	1	directional	40 Hz - 20 kHz	-38dB re 1V/Pa $\pm$ 2dB @ 1kHz

We used a quiet office room for the recordings and a printed template for data collection. Volunteers were asked to fill in the printed sheets with numbers. There were two types of templates, each containing ten rows. In the first template, the rows had to be filled with sequential numbers starting from zero and ending with nine. In the second template, the row number determined which numbers had to

be written to the corresponding row.

Testing showed that the available microphones could only capture the signal from a close range, so we decided to place the microphone next to the template sheet. More specifically, it was placed on the table so that 15 centimetres would separate the microphone from the edge of the paper sheet.

In total, we had 11 volunteers, and some of them filled in both of the template sheets. The volunteers were asked to make a small pause before filling in each cell in the sheet. This request was made so that it would later be possible to automatically split the recording into separate audio files such that each file would contain one labelled sample.

As a result of the data collection, we created a dataset, which consisted of 1676 samples. One of the recordings was corrupted, which allowed only 76 samples to be extracted.

4.2.2. Processing the dataset

Before the recordings could be split into labelled samples, they had to be manually reviewed. We used Audacity to preprocess the recordings. Each recording captured the audio of a volunteer filling in one template sheet. However, the recordings sometimes also contained loud noises and non-relevant audio, which had to be cut out. In addition, we removed the redundant parts from the beginning and end of the recording.

Next, we created a Python script to process the files. We used Pydub library [277] for silence detection to split each recording into samples. As the volunteers wrote the numbers in a predetermined manner, the script could automatically label the samples. In case the recording was split into an invalid number of samples, it was obvious that the recording contained noise, which was not removed by the silence detection algorithm. Thus, it was easy to review and remove the problematic noise from the recordings. Once such issues were resolved, the audio files were split and the resulting samples labelled.

However, that was insufficient to make the samples comparable as the volunteers were not writing the numbers at a uniform pace. Therefore, the length of the audio files that had the same label was not constant. To solve this issue, we had to time-stretch the audio files to make them comparable to each other. For that, we used the WSOLA algorithm [312] provided by the PySOX library [56]. While processing audio with WSOLA can add minor artefacts, it does not change the pitch of the sound [114, 312]. This was an important factor when choosing the suitable algorithm, as the change in pitch would distort the digit's representation. We resized the samples such that all of them would have a length of 0.55 seconds, which was close to the average length of the samples.

To classify the samples, they have to be made easily comparable. The straightforward idea is to convert them into the frequency domain by applying fast Fourier transform (FFT). However, the sample also contains the time dimension, which

carries the information about the movement of the pen on the paper. Therefore, to also capture the time dimension, we processed the samples with the help of the SciPy library [313] to create their spectrogram representations. A spectrogram is created by running FFT on the audio fragments, which results in a two-dimensional array that represents both the time and frequency domains. A visual example of spectrograms representing digits can be seen in Figure 7.

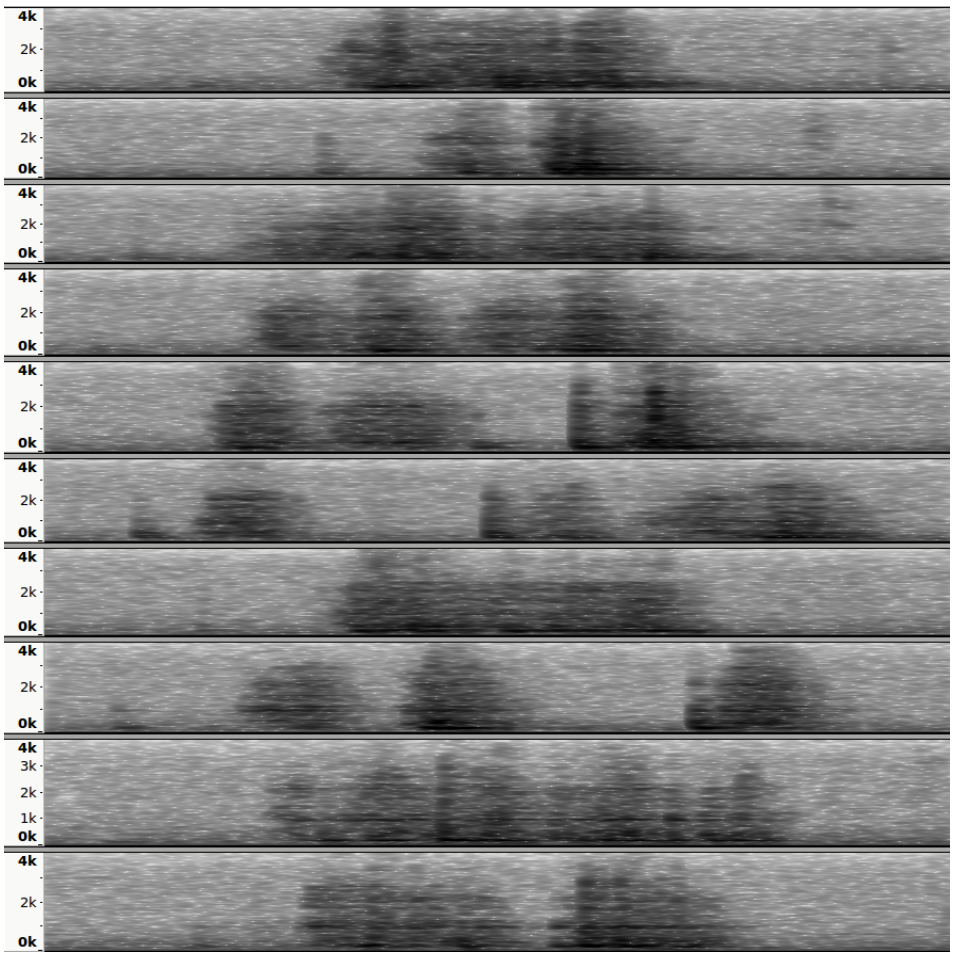


Figure 7. Spectrogram representations of sequential numbers starting from zero at the top and ending with nine at the bottom. The horizontal axis represents time.

4.2.3. Finding a suitable classifier

The limited size of the dataset discouraged us from applying advanced classification techniques that rely on neural networks. Thus, we approached the classification problem with the common k-nearest neighbors algorithm (k-NN) [99], which can produce good results even with a small dataset. The algorithm works

by finding the distance between the samples and deciding the classification based on the majority vote on the k nearest samples.

To run k-nearest neighbors, we had to flatten the arrays, which were created when the samples were converted into the spectrogram representations. This resulted in each of the samples being described by a one-dimensional array. We did not write the classification algorithm on our own; instead, we used the readily available scikit-learn [256] implementation.

Before measuring the accuracy of the classifier, the optimal parameter values had to be selected for k-NN. To do that, we split the dataset into a training set and a test set. We relied on the implementation from scikit-learn, which contains a function `train_test_split` that allows samples to be split so that the labels are distributed uniformly. Still, as the splitting was done on the dataset level, it was not guaranteed that the individuals who contributed to the dataset were uniformly distributed between these sets. We used 90 percent of the samples for training and the remaining 10 percent for testing.

Next, we had to determine how to measure the distance between the spectrogram representations of the samples. We used trial and error to determine the most suitable distance metric. It turned out that Canberra distance [204] clearly outperformed the other commonly used metrics.

Finally, we had to select the optimal value for the k parameter to measure the classification accuracy with k-NN. We selected the value for this parameter by running the classifier using different odd valued integers. More specifically, we applied cross-validation to identify that the k value that gave the best out-of-sample accuracy was 7.

4.2.4. Determining classification accuracy

The accuracy of the classifier had to be determined to check whether the acoustic side-channel-based number detection would be feasible. A good overview of the classifier's performance is given by running cross-validation, which iteratively runs the classifier on the same dataset. More specifically, the dataset is split into n equally sized non-overlapping subsets. Cross-validation is done in n iterations such that in each round, a new subset is used for validation, and the remaining ones are used for training. Thus, the parameter n determines how many iterations are done. As there were ten labels in our dataset, we selected n to be equal to ten. In addition, the ratio of labels would have to be similar in each of the subsets for the results to be comparable. This property is provided by cross-validation that uses stratified KFold partitioning.

We used the cross-validation implementation provided by scikit-learn as it applies stratified KFold by default. Running the cross-validation with the previously described configuration gave an average accuracy of 60.14%. To visualise the error rates, we also ran cross-validation predictions for each of the labels. The resulting confusion matrix can be seen in Figure 8.

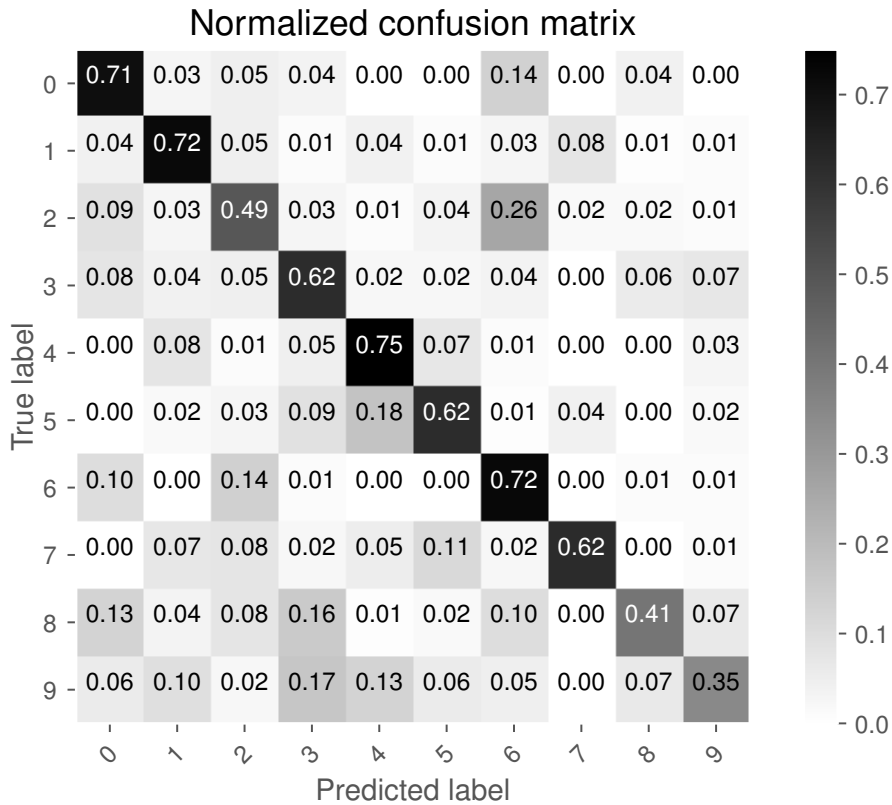


Figure 8. The image originates from [196] and contains a confusion matrix that was created from the output of cross-validation. The color scale shows the accuracy of the classification. We can see the correctly classified numbers on the diagonal of the confusion matrix. The accuracy of cross-validation was 60.14%.

The diagonal of the confusion matrix represents the correctly classified samples. It shows that only three digits had a classification accuracy that was below 50%. The label corresponding to the sound of number two being written was classified correctly in 49% of the cases. Out of the incorrect classifications in 26% of cases, number six was predicted. The other two digits that had low classification accuracy were eight and nine. There can be multiple reasons for their lower accuracy. One of these reasons may be related to the placement of the microphone when the samples were collected. Namely, the template paper was in landscape mode, and the microphone was placed next to the top middle part of the sheet. Thus, the audio signal that originated from the cells in the middle of the sheet probably had a slightly better quality. Another reason for the low accuracy of these two digits could be caused by the way how the scikit-learn's implementation of the classification algorithm breaks ties. This is done according to the ordering of the classes. For example, if there is a tie between digits three and

eight, the former would win and lower the prediction accuracy for the latter.

When considering the practical applicability of audio-based number detection, the results should be generalisable. In the context of a side-channel attack, it is unrealistic to assume that an attacker has access to personalised training data. Thus, two other measures are of interest to us: prediction accuracy in case training data is available for the same subject and prediction accuracy in case such data is unavailable.

To measure the prediction accuracies, we took one dataset from each of the eleven volunteers. Thus, each person contributed 100 labelled samples. First, we measured the classification accuracy based on personalised training data. For that, we took 10% of the stratified samples from each volunteer and used the rest of the samples for training. As expected, this resulted in significantly higher accuracy so that, on average, 70.5% of the samples were classified correctly. This result can be compared to the 65% accuracy of the handwriting detection experiments done by Yu *et al.* [324].

Finally, we measured the classification accuracy in case there was no training data available for the subject whose samples had to be labelled. To run these tests, we split the datasets of the volunteers such that ten datasets were used for training and the remaining one for validation. Thus, the training set consisted of 1000 samples and the validation set of 100 samples. We iterated this process eleven times. This resulted in an average classification accuracy of 49%, with the best case having an accuracy of 65% and the worst case 37%.

Testing revealed that by increasing the size of the training set, the average accuracy gradually increased. Thus, it might be possible to improve the accuracy by getting training data from a larger set of volunteers. However, this may also have the opposite effect if the volunteers who contribute the data have very different cultural backgrounds. Namely, we found out that one of our potential volunteers used a completely different technique for writing numbers due to having a different cultural background. Therefore, the cultural background also has to be taken into account when creating such a dataset.

4.2.5. Discussion

The conducted experiments showed that, in principle, it is possible to use the audio channel to identify the digits written on a piece of paper. However, a proof-of-concept implementation does not guarantee that such an attack would work in real-life conditions.

A voting booth in a polling station is likely to have a significant amount of background noise, which would have to be removed from the recordings. This also means that the microphone would have to be very close to the audio source as otherwise the sound emitted by the pen would not be captured. We see that there are two ways how an attacker could solve this problem. One option would be to place the microphone under the voting booth table. It would still be possible

to capture the sound as the table plate vibrates when the pen touches its surface, thereby making it possible to measure the signal. Another more theoretical option would be to use a laser microphone to capture the sound remotely [225]. However, a laser microphone would require a direct line of sight to a reflective surface close to where the sound is emitted, but in case a direct line of sight is available, a camera could be used instead.

The possibility of using such a side-channel depends on the design of the ballot sheet. There are a few countries where candidate numbers are directly written to the ballot and some countries where candidates have the be enumerated. In the former case, the candidate numbers usually consist of several digits, but not all combinations of digits are assigned to candidates. In addition, the predictions from the classifier could be used to eliminate a large set of candidates, which the voter very likely did not vote for. An attacker could use such auxiliary information to improve the accuracy of an attack that targets vote privacy.

In the case of preferential ballots, the exploitability of the side-channel depends on the structure of the ballot, how the ballots are printed and how voters are supposed to fill in the ballots. For example, it may be possible to detect donkey voting, in which case the voter ranks the candidates according to their position on the ballot [250]. However, suppose the candidates are printed to the preferential ballot sheets in random order. In that case, it becomes impossible to exploit the acoustic side-channel as the attacker cannot connect the candidate names to the choices made by the voter.

There are two main options for limiting the risk from acoustic side-channel leakages. First, it should be made difficult to hide microphones in the voting booths and the voting booths should be inspected to make sure that there are no hidden microphones. Second, in principle, it is possible to change the design of the ballot sheet to reduce the amount of information that can leak. However, as the likelihood for such an attack is rather low, it is unlikely that the design of the ballot would be changed just to reduce the leakages from the acoustic side-channel.

4.3. A side-channel against paper voting – mark detection

The amount of information that could be leaked through an acoustic side-channel heavily depends on the design of the ballot. For example, only a few countries use ballots where the candidate numbers have to be written on the ballot sheets [272]. Instead, voters are often asked to show their preference by marking the candidates listed on a ballot. In such cases, the acoustic side-channel that allows reconstructing digits does not apply as the voters mark checkboxes instead of writing numbers.

When checkboxes are used to mark the preferences, all candidate names have to be printed on the ballot sheet. However, in some elections, there can be hundreds or thousands of candidates. On these occasions, the ballot sheet has to be quite large to fit all of the names.

Examples of such candidate lists can be seen in Figures 5 and 9, which depict ballots that have been used in some German and Dutch elections. Both of these list hundreds of candidates. Ballots with similar sizes have also been used in Australia, e.g., for the Victorian Senate elections in 2013 [77].

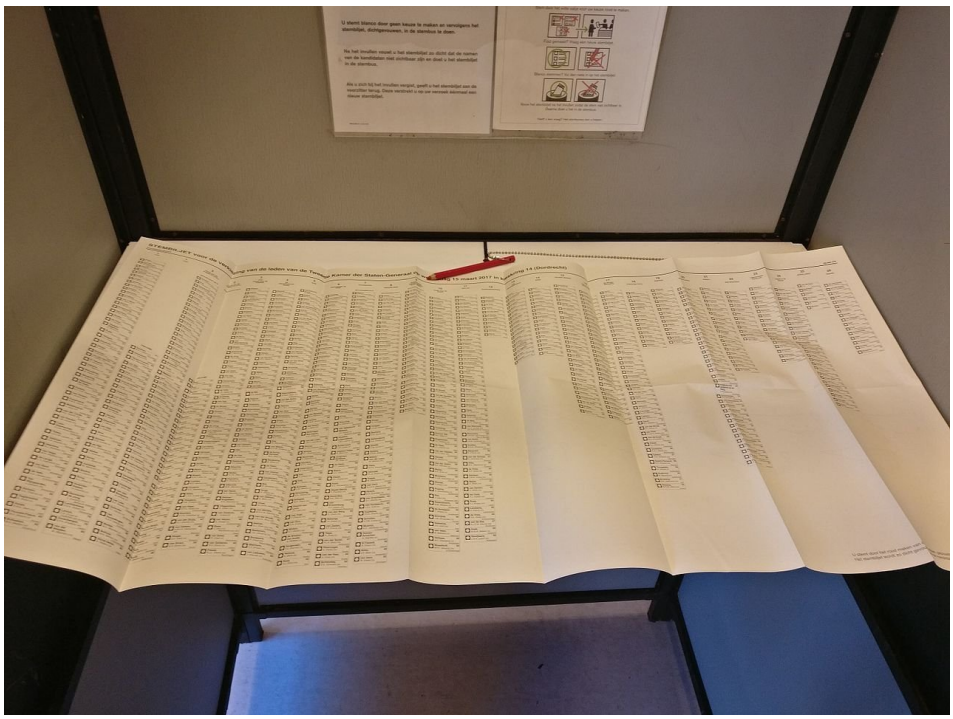


Figure 9. Dutch general election ballot sheet from 2017. Image author: 1Veertje, licence: CC BY-SA 3.0, source: `https://commons.wikimedia.org/wiki/File:Ballot_2017_Dutch_general_elections_-_1.jpg`.

The large ballot sheets that cover most of the voting booth table lead to a new side-channel leakage. Although the sound of filling the checkboxes does not directly reveal anything about the vote, it turns out that the source of the sound gives information about the voter's selection. For example, the Dutch ballot sheet depicted in Figure 9 lists party candidates in columns. Such a design allows determining the location where the sound is emitted to predict which party the voter voted for. More importantly, the information could be used to eliminate a majority of the candidates and parties from the possible choices that the voter made.

These observations led us to the question of how feasible it is to implement a proof-of-concept attack against the privacy of paper-based voting. It turned out that with suitably placed sensors, it is possible to determine the origin of the signal and use it as a side-channel [197]. An overview of the experiments that we conducted while investigating the side-channel is given in the next sections.

4.3.1. Finding an optimal setup

Multiple sensors are required to pinpoint the origin of the signal. We had already observed in the previous experiments that the sound of a pen scratching the surface travels in the table plate through vibrations in the material. Therefore, we knew that it should be possible to use sensors attached to the corners of the table plate to capture the signal of the pen or pencil touching the ballot paper.

Our initial idea was to attach multiple microphones to the table to locate the signal by using time difference of arrival (TDOA) [292, 151]. Thus, we created a setup, which allowed us to conduct experiments. As we did not know how the structure of table material would affect the results, we decided to build two tables out of different materials. One of the table plates was built from melamine-covered chipboard, and the other was from glued timber. Both had dimensions of 80×60 cm.

Next, we needed to find suitable sensors to capture the signal. Literature review revealed that piezoelectric elements might work in such a scenario [271]. However, we only had access to cheap piezoelectric elements, which could not capture the signal.

Before purchasing more expensive sensors, we decided to test the approach that relied on microphones. It turned out that the cheap electret microphone amplifier MAX4466 could capture the signal when attached to the table plate.

We used Arduino Due as the computing platform due to the processing power of its ATMEL SAM3X8E ARM Cortex-M3 CPU, which ran at 84 MHz. The scheme for the microphone connection to Arduino Due can be seen in Figure 10.

We had to use at least three microphones to identify the location of the signal. However, we decided to include also the fourth one in our setup as it allowed for their symmetrical placement to the corners of the table. While it would have been possible to add more microphones to increase the accuracy of locating the signal's origin, such an approach would also have its downsides. Namely, there is only one analog-digital converter (ADC) in Arduino Due, which prevents the signals from multiple microphones from being processed in parallel. This creates a time delay before the ADC can process the signal from the next microphone. Thus, by increasing the number of microphones, it takes more time for the emitted audio signal to be read by all of the microphones, which may result in a decreased accuracy. Fortunately, the ADC does not have to wait between the conversions as it can work in the free-running mode, which allows starting the next conversion right after the previous one has finished. For our device, the free-running mode had a working frequency close to 600kHZ.

Next, we had to attach the Arduino Due and the microphones to the table plate. As we aimed to build a proof-of-concept device to measure the information leaked through the acoustic side-channel, we did not conceal the device. Instead, we decided to tape our setup to the table plate.

There are two possibilities for attaching the microphones. First, it is possible

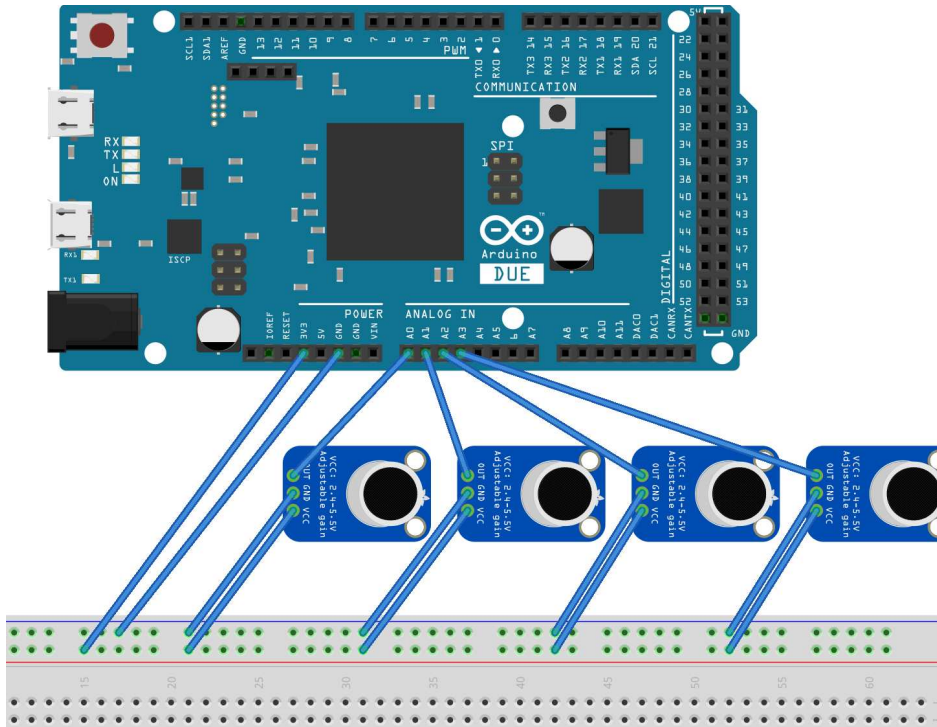


Figure 10. Connection scheme for Arduino Due and four MAX4466 microphone amplifiers. The image originates from [197].

to tape them to the surface of the table plate. Second, the microphones could be placed into holes that are specially drilled into the table plate. The latter approach would provide cover for the microphones, which would both hide the devices and possibly also reduce the amount of background noise. In addition, we thought that such a placement would allow for better contact with the table plate, which would help to pick up faint signals.

We decided to test both approaches. Thus, the first experiments were performed with the microphones taped to the surface of the table plate and the following experiments with drilled-in microphones. The microphone on MAX4466 has a diameter close to 1cm, while the circuit board has dimensions of 1×2cm (see Figure 11). Therefore, we decided to drill a hole only for the microphone and not hide the circuit board of MAX4466. Although one of our motivations was to increase the quality of the captured signal, the following experiments did not show such results. It turned out that the table plate materials made it difficult to place the microphones in the drilled holes so that the whole microphone would touch the table plate.

Finally, the experimental setup consisted of a custom-built table and four electret microphone amplifiers connected to Arduino Due based development board. The setup can be seen in Figure 12.

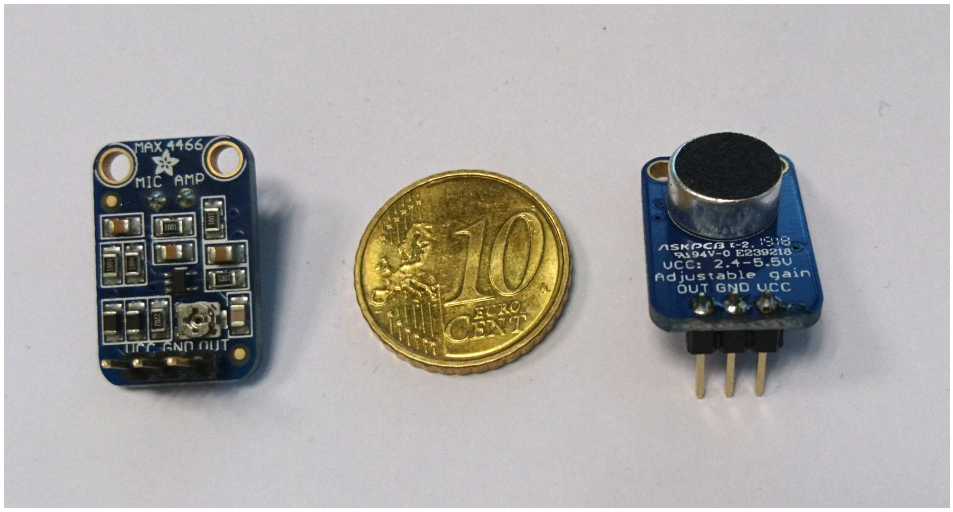


Figure 11. Both sides of the MAX4466 electret microphone amplifier are placed next to a 10 cent coin. This image originates from [197].

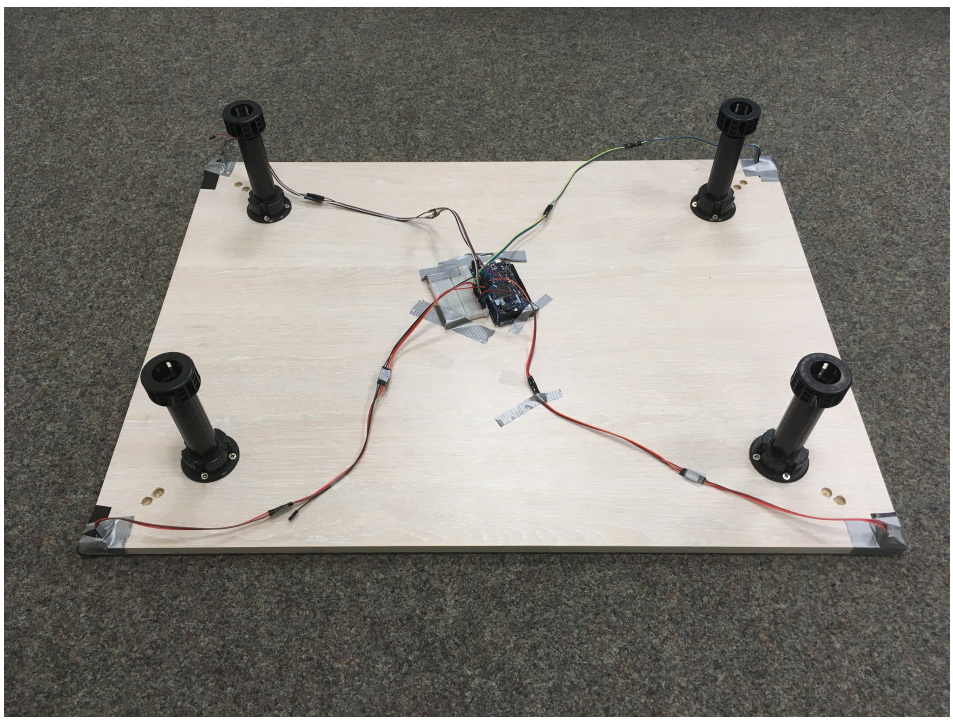


Figure 12. Experimental setup from below (melamine-covered chipboard). The image originates from [197].

4.3.2. Determining the origin of the sound

Using the experimental setup, it was possible to figure out how the captured data could be processed. The signal coming from the microphone shows the voltage level at the time of the measurement [16]. However, as the microphone is continuously listening, there is always some background noise. So the first task was to isolate the signals that we wanted to capture. To do that, we decided to compare the output from the microphones with a reference value that acts as a threshold for identifying a sufficiently strong signal.

We considered two methods for determining the origin of the signal. As the first option, we considered using signal strength as it is likely to correlate with the proximity of the microphones. If this is the case, measurements should show that the signal's strength decreases with the increase of the distance from the location where the sound was emitted. We planned to capture these differences by measuring the total energy of the signals for each of the microphones. The energy of the signal $x(t)$ can be calculated with the formula

$$\int_{-\infty}^{\infty} (x(t))^2 dt \ ,$$

which can be approximated by

$$\sum_{i=0}^{N-1} (x(i))^2$$

where i marks the reading and $x(i)$ the amplitude of the sampled signal during the i-th reading.

The other option for locating the origin of the signal is to measure the time difference of arrival (TDOA). The measurement is triggered by the first microphone that captures the signal. That time-point is taken as a baseline to determine how much time it takes for the signal to reach each of the other microphones.

Although the information gathered by measuring TDOA could be used to find an analytical solution, it requires knowledge about the speed of the signal. However, the speed of sound in wooden materials varies. Furthermore, if the structure of the material is non-uniform, the speed of sound may differ depending on the direction the signal moves. Thus, instead of finding an analytical solution, we decided to collect test data so that it would be possible to approach the problem with classification algorithms.

We ran experiments and concluded that measuring the TDOA outperformed measuring the amplitude of the signal. We are not fully sure what were the reasons for this, but one of the guesses is that the signal might reflect from the edges of the table plate [278] and thereby add noise. Importantly, this type of problem does not significantly affect the approach that is based on measuring the time delays.

4.3.3. Data collection and classification

Once the optimal method for pinpointing the signal's location was determined, the next step was to measure its accuracy. For that, we needed to collect labelled test data so that each data point would consist of the readings of all of the microphones along with the coordinate where the sound was emitted. Thus, we had to select a coordinate system for the table plates while considering the physical limitations like the speed of sound.

The speed of sound in wood is approximately $3 \ldots 6\,\mathrm{km/s}$ depending on the type and density of the wood, its moisture level, and the movement direction [311, 38]. However, wood-based composite materials have slightly lower speeds, which usually fall into the range of $2 \ldots 3\,\mathrm{km/s}$[46]. In addition, the structure of the composite materials is more uniform compared to wood planks, which causes the sound velocity to be less dependent on its movement direction.

The ADC in our setup works with a frequency close to 600kHz, but the four sequentially read microphones have to share the ADC. Thereby, each of the microphones is read with a frequency close to 150kHz.

Based on that information, it is now possible to calculate the minimum distance that can be measured between two consequent reads with our setup. To find an estimate for the minimal measurable distance value, we assumed that the speed of sound in wood-based composite materials is equal to $3\,\mathrm{km/s}$, which results in a distance of

$$\frac{3\frac{\mathrm{km}}{\mathrm{s}}}{150000\frac{1}{\mathrm{s}}} = \frac{3000\mathrm{m}}{150000} = 2\mathrm{cm}\,.$$

Due to the uncertainty of the speed of sound in our setup and also to cover for additional measurement errors, we decided to split the table plate into 4×5 cm cells. Thereby, the 80×60 cm plate was divided into a 20×12 grid, which we used as a basis for determining the coordinates.

As the setup was complete, we could start collecting test data. For that, we mimicked the process of using a pencil to mark a field on a ballot sheet. As a first approach, we wrote the marks on a paper sheet that was placed in a suitable place on the table plate. By looking at the measured time delays, the signal quality seemed to be sufficient. However, it turned out that using a paper sheet is not the optimal way to capture thousands of data points. Moving the paper sheet thousands of times would hurt the precision, add noise, and slow down the data collection. Thus, to simplify the process, we decided to write the marks directly to the table plate.

We drew the grid to the table plate, resulting in 240 cells. Each time a cell was marked, the microphones captured the corresponding timing information. More specifically, for each such event, we captured the values t_1, t_2, t_3, t_4, which represent the times when each of the four microphones detected the signal. As our goal was to compare the time delays, we were interested in the time deltas with

respect to the time when the first microphone picked up the signal. Thus, for each data point, we stored the following tuple $(t_1 - t, t_2 - t, t_3 - t, t_4 - t, x, y)$, where $t = \min\{t_1, t_2, t_3, t_4\}$ and x, y represent the coordinates of the cell in the table plate.

In total, $10 \ldots 15$ marks were made to the centres of each of these cells. The resulting table plate can be seen in Figure 13.

Figure 13. Cells and markings on a table (glued timber plate). The image originates from [197].

Once we had the data, we were able to begin with the classification task to predict in which cell the data point was created. We tried out five classification algorithms: k-Nearest Neighbour (k-NN), a weighted version of k-NN, Gradient Boosting Classifier, Multi-Layer Perception Classifier and Random Forest Classifier [124]. It turned out that k-NN-based classifiers gave the best accuracy. An overview of the results is given in Table 4.

Table 4. Accuracy of the classifiers which were tried out while building an acoustic side-channel in [197].

Method	Accuracy
Weighted k-NN	90.4%
k-NN	89.2%
Random Forest Classifier	87.3%
Gradient Boosting Classifier	84.6%
MLP Classifier	16.3%

To run k-NN, we had to select a suitable distance metric and the value for the parameter k, which determines how many neighbours are considered when classifying a new data point. Based on the structure of the data points, we decided to compare the following three distance metrics: Canberra, Euclidean, and Bray Curtis. Testing revealed that the Bray Curtis metric resulted in the highest prediction

accuracy when used with $k = 5$.

Given two data points $u = (u_1, u_2, u_3, u_4)$ and $v = (v_1, v_2, v_3, v_4)$, their Bray Curtis distance is defined as

$$d_{BC}(u,v) = \frac{\sum_{k=1}^{4} |u_k - v_k|}{\sum_{k=1}^{4} (|u_k| + |v_k|)} .$$

For the weighted version of k-NN, weights have to be set. These were defined in the form of $1/d^e$, such that d is the selected distance metric and e represents the weight. Before determining the optimal value for the weight exponent, we experimented with values of e being taken from the set $\{1,2,3,4,5,6\}$.

The initial step of weighted k-NN works similarly to regular k-NN as the new data point v is compared to the values in the training set according to the selected distance metric. This results in k closest data points $u^1, u^2, \ldots, u^k$ being identified.

As a next step, the data points are grouped according to their labels, which in our case means the cell coordinates. In the case of weighted k-NN, the weights are determined for the found data points according to the formula

$$\sum_{j=1}^{n_i} \frac{1}{(d(u^{i,j}, v))^e} ,$$

where u^i denotes the data point, which belongs to group i and e is the weight exponent. Thus, $\{u^{i,1}, u^{i,2}, \ldots, u^{i,n_i}\}$ is the set of tuples in the ith group, which means that they all have the same label. After calculating the weights for different groups, the group with the largest weight is selected as the winner. Thus, the label associated with the winning group is assigned to the data point v that was being classified.

4.3.4. Prediction accuracy

Having found the optimal classification algorithm, the next step was to evaluate the accuracy of predicting the coordinates for the data points.

Our setup consisted of two table plates, which were made of different composite wooden materials. We had the option to either tape the microphones to the surface of the table plate or drill them in. This resulted in four possible combinations for the setup, and we collected test data for each of these.

To evaluate the applicability of such a side-channel, we wanted to know how accurately could the origin of a recorded signal be determined. As the candidates from the same party are commonly placed into the same column in the ballot sheet, we were interested not only in determining the correct cell but also the correct column. Thus, we measured the accuracies of

1. predicting the correct 4×5 cm cell,
2. predicting the correct column,
3. predicting the area that includes the correct cell and the eight neighbouring cells (i.e. 3×3 cells that form a 12×15 cm rectangle).

We used 10-fold cross-validation to measure the accuracies, which meant that the dataset was iteratively distributed into 10 subsets. Nine of these were used for training and the resulting one for validation. The experiments were run 500 times. The resulting average accuracies are presented in Table 5.

Table 5. Prediction accuracy for the experiments that were used to evaluate the acoustic side-channel for mark detection [197].

Test setup	1×1 cell	3×3 cell	Column
Chipboard, drilled mics	90.44%	98.77%	93.71%
Chipboard, taped mics	81.07%	94.14%	85.23%
Glued timber, drilled mics	71.29%	87.89%	75.42%
Glued timber, taped mics	77.61%	89.62%	80.35%

We can see from Table 5 that the highest accuracy was achieved with a chipboard-based table plate that had the microphones drilled in. The prediction accuracy of this setup was close to 90% when considering the exact location of the signal. The same prediction for the table plate made of glued timber had significantly lower accuracy, which was probably caused by the less uniform structure of the wooden planks that were glued together.

Interestingly, with the glued timber plate, the accuracy was higher if the microphones were taped to the surface of the table plate. The reason was probably in our inability to drill flat-surfaced holes due to the structure of the wood and also due to the shape of the used drill bit. Thus, the microphone was probably not fully in contact with the surface of the plate.

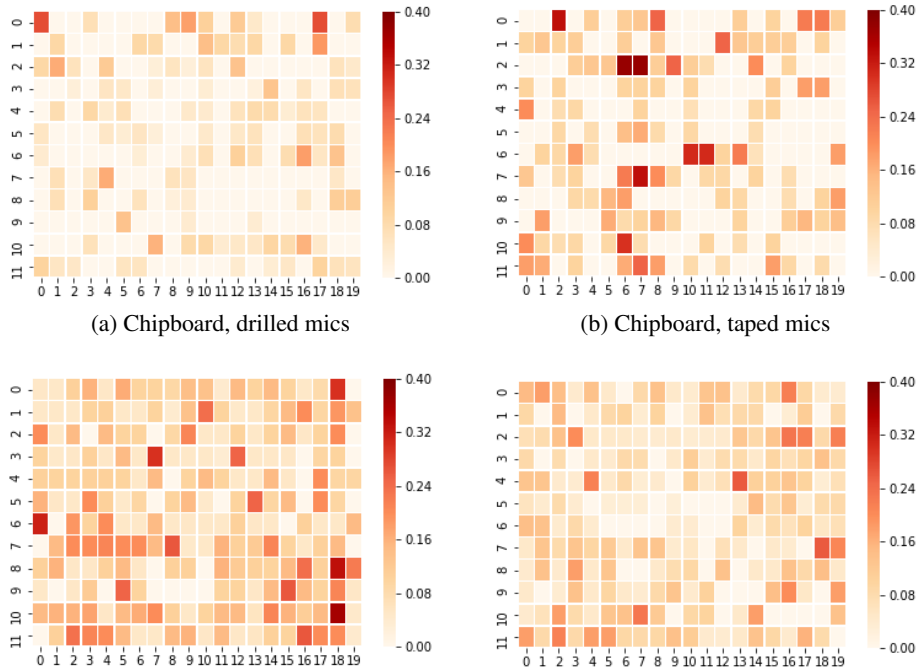
Finally, we measured the error rate of our predictions to see whether the prediction accuracy is uniformly distributed in the cells on the table plate. For that, we calculated the ratio of incorrect classifications and used heatmaps to visualise the results. The heatmaps displayed in Figure 14 do not seem to show systematic classification errors.

4.3.5. Discussion

The experiments showed that, in principle, it is possible to reconstruct the location where the voter marks the ballot sheet. However, this does not automatically mean that such an attack is practical.

First, the microphones would have to be placed into the voting booths, which requires physical access. Second, the recording equipment would have to be stealthy so that the voters would not detect it. Third, the noise level in the real world may make the attack impractical. Fourth, the voter would have to be identified to tie an identity to a vote. Fifth, the risk could be probably mitigated by building the voting booth tables out of materials that are bad at carrying sound waves.

Although our study did not consider all of the real-world scenarios, we can make some educated guesses. For example, when considering how to place and



(a) Chipboard, drilled mics

(b) Chipboard, taped mics

(c) Glued timber, drilled mics

(d) Glued timber, taped mics

Figure 14. The heatmaps illustrate the error ratio distribution across the experiments. The axes represent the coordinates of the 80×60 cm plate, which was divided into a 20×12 grid. The lighter coloured cells represent low error ratios, while the darker coloured cells represent higher error ratios. The image originates from [197].

hide the microphones, there are two main attack scenarios. One involves an insider who has access to the voting booth, which significantly simplifies the deployment of recording equipment. On the other hand, an external attacker would have to modify the voting equipment either while it is in storage or after it has been deployed to the voting station. The latter requires access either during off-hours or the ability to attach the microphones while election officials are present. This process is somewhat simplified if the voting booth contains a curtain that hides what is being done inside the booth.

It is unclear how much the noisy environment would affect such an attack. This could be measured by setting up a study, which involves asking people to vote in mock-up elections. However, noise reduction would likely have to be applied for the attack to work.

The issue related to identifying voters once again depends on who is the attacker. In the case of a malicious election official, it may be possible to use the information provided by the voter to connect the name to the vote. However, an external attacker would have to either monitor the voting station or find a way to automate this process. A small-scale coercion attack may work even when a single attacker monitors the people who enter the voting booths. To scale up the

attack and observe multiple voting stations, the attacker would have to hire more observers or use technology for this task.

One option is to use face detection to check which voters enter the voting booths. However, there are multiple difficulties in using this approach in practice. For example, an attacker would need to have a camera in the voting station. Interestingly, it turns out that sometimes it can be trivial to capture such information in case web cameras are placed in the voting stations with the aim of increasing transparency [229, 170]. Still, a camera on its own is not sufficient to match faces with names as it is unlikely that a coercer can access a database that contains high-quality images of voters.[2] Although, this assumption may not hold in case the election organiser is involved in the attack. However, an attacker who plans to buy votes may ask the willing voters to upload their photographs. It may also be possible to scrape social media profiles to gather voters' photographs. Our experiments have revealed that once reference photos are available, facial recognition can be implemented by using simple, readily available components [197].

Unsurprisingly, it turns out that technology can be applied to attack regular paper-based voting systems, although they are often considered superior to electronic voting systems. Furthermore, as it has become possible to use smartphones and other devices to record the vote casting process, it is questionable whether a voting booth can provide privacy for a coerced voter [48].

The aforementioned attacks against vote privacy once again highlight the need to re-evaluate the security of existing voting systems. The proof-of-concept acoustic side-channel attacks targeted the vote privacy of paper-based elections, but these are not the only issues with existing voting systems. For example, the lack of cryptographic integrity checks puts pressure on election observers to guarantee that the results are not tampered with. However, modern cryptography makes it possible to improve integrity guarantees of voting systems, including paper-based voting systems [281]. One of the reasons why such voting protocols are not used by default lies in the conflict between vote privacy and integrity guarantees. If the legislation requires votes to remain secret, it becomes questionable whether end-to-end verifiable voting systems can be implemented. The advancement of technology further complicates this issue and adds another controversy. As it is possible to record the vote casting process in the voting booth, absolute vote privacy can not be achieved, at least not when the voter is willing to record the vote. Thus, it has to be analysed whether implementing verifiability measures would introduce additional privacy leakages compared to privacy leakages in existing voting systems.

We can see that achieving absolute vote privacy and coercion-resistance is in-

[2] As a counterargument, it could be said that the photos may be bought from hackers. For example, it was revealed in 2021 that a hacker downloaded close to 280 thousand document photos from an information system belonging to the Estonian state [318]. Similarly, a hacker managed to breach the Argentinian governmental database in 2021 and copy information related to national identity cards, including document photos [72].

feasible in practice. Thus, some level of coercion has to be considered acceptable. This raises the question of what is the acceptable level of privacy and coercion-resistance that the voting system must provide. For example, one reason why end-to-end verifiability is sometimes disregarded when considering remote voting systems is that it increases the risk of coercion and vote buying. However, the coercion risks introduced by end-to-end verifiability may not significantly differ from the risks in existing voting systems, especially if postal voting is considered. If the election organisers accept that absolute vote privacy and coercion-resistance can not be provided, it may lead to end-to-end verification being introduced also for paper-based election systems. For example, in 2021, Benaloh described how verification could be added to postal voting [49].

Both in paper-based voting systems and electronic voting systems, one of the main threats comes from insiders. By introducing end-to-end verification, the insider threat to the vote integrity would be significantly reduced, while voters would be given an opportunity to prove how they voted. In general, it is important to prevent large scale coercion attacks, regardless of whether the vote privacy is violated in voting stations, via malware, or by the voters themselves by abusing the functionalities offered by the voting protocol.

5. IMPROVING VOTING PRIVACY AND VOTE INTEGRITY IN THE ESTONIAN I-VOTING SYSTEM

In this chapter, we give an overview of the i-voting system used in Estonia, describe the client-side issues of the Estonian i-voting system, and analyse the possible mitigation measures. We show that some of the client-side risks can be reduced by creating a microcontroller-based voting application. Finally, we analyse the security aspects of introducing a smartphone-based voting application.

5.1. Overview of i-voting in Estonia

The Estonian i-voting system has been in continuous development since the beginning of the first decade of the 21st century. It has its origin in the supporting legislation, which led the path to the adoption of i-voting. Passing of the Digital Signatures Act in 2000 laid the groundwork for introducing ID cards [4]. In 2002, Riigikogu Election Act was passed and, among other things, it mentioned vote casting over the internet [7]. In 2005, legislation was passed that allowed i-voting to be used for the municipal elections [8] and in 2006, Riigikogu Election Act was amended such that i-voting could be used for parliamentary elections in 2007 [9].

In 2001, two analyses were published that covered the feasibility of implementing an i-voting system in Estonia. One of these analyses concluded that, in principle, it was possible to introduce i-voting already in 2002 [304]. The other analysis recommended doing research in the field of i-voting so that i-voting could be introduced in four to eight years [212].

In 2003, Estonian National Electoral Committee (NEC) initiated a project to introduce i-voting [126]. As a part of the project, an i-voting scheme was created [83, 84]. In addition, a security analysis was published, which stated that an optimal solution had to be found between the targeted security level and implementation complexity [24, 164]. As a compromise, the scheme relied on strong trust assumptions. More specifically, the security analysis stated the following (translated from Estonian): “The other side of the compromise or, in principle, the weak point of the scheme, is the need to trust central servers and computers of the voters. Is such a compromise reasonable? In our opinion – yes.” [24, 164].

I-voting was first used in Estonia during the municipal elections in 2005 [164]. Starting from 2007, it has been used for parliamentary elections and since 2009 also for the European Parliament elections. Over the years, i-voting has become one of the main vote casting channels. While in the municipal elections in 2005, only 1.9% percent of votes were cast over the internet, in the municipal elections in 2021, the percentage rose to 46.9% [93].

Estonian i-voting system has strong foundations in electronic identities (e-IDs) and e-services provided by the Estonian state. The Identity Documents Act states that an ID card is mandatory for Estonian citizens who are at least 15 years old [3].

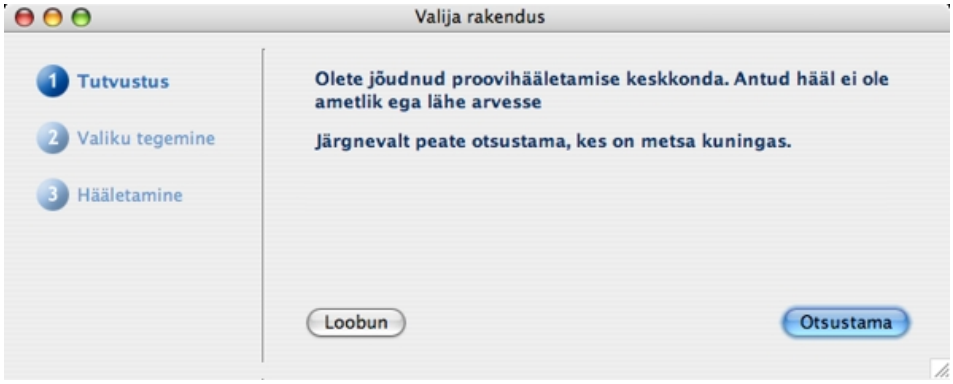


Figure 15. A screenshot of the Estonian i-voting application used in 2007. The screenshot displays a voting application that is configured for mock elections. Image source: archived website of the Estonian National Electoral Committee [81].

The usage of ID cards made it possible to reliably authenticate voters.

ID card is a smart card-based identity document issued by the state. It can be used to authenticate the card owner digitally and to issue legally binding digital signatures. The chip of the ID card contains two key pairs, one for issuing legally binding signatures, and the other one is mainly used for authentication. By default, the card is issued with two valid certificates, one for each key pair. However, the certificates can be revoked if the card owner does not want to use the electronic functionalities provided by the ID card.

In addition to the ID card, it is possible to acquire the following non-mandatory electronic identities: Mobile-ID, digital identity card (digi-ID), e-resident's digi-ID, diplomatic ID, residence permit card and Smart-ID[1] [36]. These electronic identities can be used to issue legally binding signatures and digitally authenticate the owner of the e-ID. While it has been possible to use the ID card for i-voting since 2005, this has not been the case for Mobile-ID. The option to use Mobile-ID to cast a vote was introduced for the parliamentary elections in 2011 [138]. As of 2021, the Estonian i-voting system supports the ID card[2] and Mobile-ID for authenticating the voters and signing the ballots.

Out of the e-services and infrastructure provided by the Estonian state, a few are required for the i-voting system to function. For example, national public key infrastructure (PKI) forms the basis for electronic identities, making it possible to issue digital signatures and identify voters. In addition, an e-population register is used to create and update the list of eligible voters. Starting from the local municipal elections in the autumn of 2021, a new election information system

[1] Smart-ID is not state-issued. As of 2021, it has not been possible to use Smart-ID for i-voting.

[2] In some cases, other electronic identities can also be used. For example, from the technical point of view, the digi-ID and residence permit card function as an ID card. Thus, if a permanent resident is eligible to vote, the residence permit card can be used for authentication and signing the ballot.

(VIS3) is used. Among other things, it unifies the handling of voter lists between polling stations and the i-voting system by replacing paper-based voter lists with an electronic version [321].

The software used for i-voting has been significantly modified over the course of time since i-voting was introduced. Between 2005 and 2009, one of the voting clients was based on ActiveX component running in Internet Explorer [80, 82, 32]. However, as Internet Explorer lost its dominant market share, it was replaced with a stand-alone voting application that the voter had to download from the election website. For MacOS and Linux users, this had already been the case since 2005 [79, 78]. The user interface of the stand-alone voting application has seen minor changes since i-voting was introduced. This is illustrated by Figures 15, 16, 17 that display the voting clients used in 2007, 2010, and 2021.

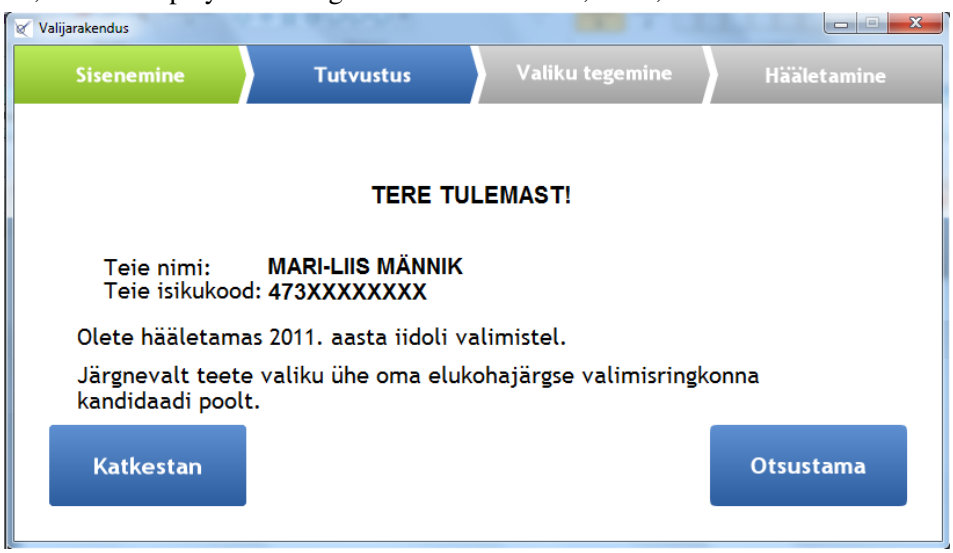


Figure 16. A screenshot of the Estonian i-voting application used in 2011. The screenshot displays a voting application that is configured for mock elections. Image source: archived website of the Estonian National Electoral Committee [85].

The first major change to the Estonian i-voting system was introduced in 2013 by adding smartphone-based individual verifiability [169]. The smartphone-based verification application allows the voter to scan the QR-code displayed by the voting application after the vote is cast. By using the information read from the QR-code, the verification application downloads the voter's ballot and displays the candidate's name.[3] The motivation to add individual verifiability originated from the parliamentary elections in 2011, which was subject to a proof-of-concept attack that aimed to highlight the issues related to untrusted end-user devices [169].

The next major upgrade to the Estonian i-voting system (codenamed IVXV) was introduced in 2017, resulting in the server-side of the i-voting system being

[3]More detailed information about the verification application is given in Section 5.1.1.

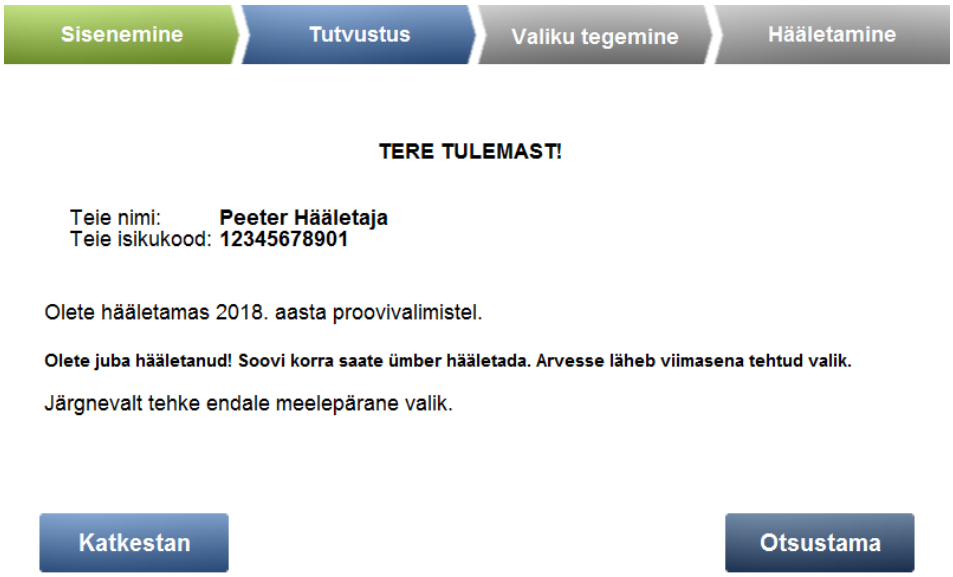


Figure 17. A screenshot of the i-voting application that illustrated the guidelines for the elections held in 2021. In practice, the voting application used in 2021 was almost identical to the one shown here. Image source: archived website of the Estonian National Electoral Committee and the State Electoral Office [92].

rewritten [165]. The upgrade aimed to add verifiability to the back-end of the voting system such that the integrity of elections would not depend on procedural security measures. While the changes in 2013 were motivated by the proof-of-concept attack conducted in 2011, the changes in 2017 were partially motivated by the security analysis of the Estonian Internet Voting System by Springall *et al.*, which was published in 2014 [293].

The upgrade in 2017 added four significant changes to the back-end system. First, an independent registration service was introduced to register all received ballots to an append-only private bulletin board. Second, a re-encryption mix-net was added to the voting system to separate identities from encrypted ballots while issuing a cryptographic proof that the ballots were not modified. Third, verifiable decryption was added, which makes it possible to cryptographically verify that decryption was performed correctly. Fourth, the private key for the elections was threshold secret-shared between several trustees.[4]

The functionalities that were added in 2017 allow external auditors to verify that no votes are added or removed by the back-end system. In addition, it can be cryptographically verified that all valid registered votes are correctly counted. As the usage of a mix-net made it possible to anonymise the ballots, it was no longer

[4]Before 2017, the election private key was generated and stored in an HSM [84]. Several National Electoral Committee members had to use their keycards to activate private key operations in the HSM.

required that the ballots be decrypted in an HSM,[5] and instead, it was permitted to use an air-gapped computer [244].

5.1.1. Architecture of the Estonian i-voting system

The Estonian i-voting system consists of client-side components and server-side components. The client-side is composed of a voting application and a verification application. The main components of the back-end are the following: the vote collector service, the vote registration service, the vote processing application, the mixing application, the key application (used in the vote-counting server), the audit application, the vote signing service, and the voter identification service [90]. An overview of the Estonian i-voting system's architecture is depicted in Figure 18.

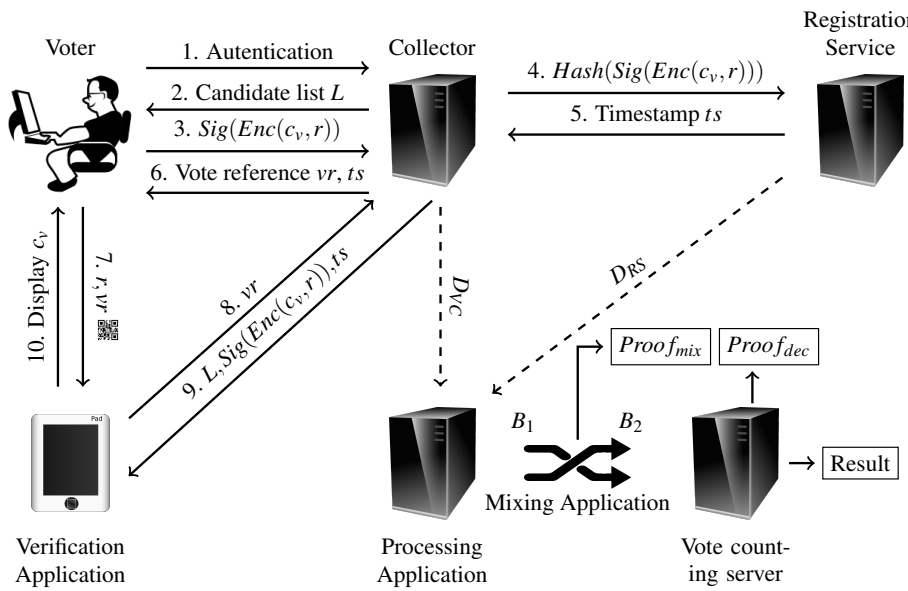


Figure 18. The figure gives an overview of the Estonian i-voting system, which has been used since 2017. The vote is denoted by c_v and randomness used for ElGamal encryption by r. The dashed arrows describe the processes that happen once the voting period has ended. In these processes, the collector and the registration service deliver the signed ballots (D_{VC}) and the signed timestamps (D_{RS}) to the processing application. More information about the depicted system can be found from [243, 169, 165]. The illustration is based on the figure from [33].

The general architecture of the Estonian i-voting system can be compared to the double envelope system used for advance voting in case the vote is cast outside the polling district of the voter. In the double envelope system, the voter is

[5] Another likely reason is that the HSM-s did not support all of the cryptographic operations used in IVXV.

first identified in person with the help of a state-provided identity document and then given a ballot sheet along with two envelopes. The voter privately fills in the ballot, inserts it into a blank envelope and seals it. Next, the voter inserts the blank envelope containing the ballot into the second envelope, which contains information about the voter, and seals the envelope. Finally, the resulting envelope is inserted into the ballot box. Once the voting period is over, the ballot box containing the advance votes is opened, the outer envelopes removed, and the anonymous blank envelopes mixed. After mixing the blank envelopes, the connections to the voters' identities should be removed, and thus the corresponding ballots can be opened and tallied.

A similar process is happening also in the case of i-voting. First, the voter downloads the voting application and runs it on a PC. The communication between the voting application and the voting system's servers is protected by TLS.[6] The voting application asks the voter to use an e-ID for authentication. Once it succeeds, the candidate list is displayed,[7] and the voter can choose who to vote for. After the choice is made, the vote (c_v) is encrypted with ElGamal using encryption randomness r generated by the voting application, and the election-specific public key, which is integrated into the voting application. The resulting cryptogram $Enc(c_v, r)$ can be compared to the anonymous inner envelope from the double envelope system. To confirm the choice, the voter is asked to use an e-ID to sign the encrypted ballot $Enc(c_v, r)$. The voter's digital signature represents the outer envelope from the double envelope system.

Next, the encrypted and signed ballot $Sig(Enc(c_v, r))$ is sent to the vote collector service, which registers the vote with the help of an external registration service that is independent of the rest of the back-end components. More specifically, the vote collector hashes the ballot $Sig(Enc(c_v, r))$ and signs the result, which is then sent to the registration service [160]. The registration service stores the hash, adds a timestamp to the hash and signs it. The resulting signed value (timestamp) is returned to the collector, who delivers it to the voting application. The voting application uses the timestamp to verify that the registration service registered the ballot.

Each ballot is tied to a vote reference (vr). The vote reference can be used in a limited time frame after the vote has been cast to query the ballot from the vote collector. This allows voters to verify their vote to check whether the cast vote was correctly received and recorded in the voting system. Once the vote has been cast, the voting application displays a QR-code that can be read by a special smartphone-based verification application.[8] The QR-code encodes the vote reference (vr) and the randomness (r) used for ElGamal encryption. If the voter

[6]The certificates of the servers with which the voting application communicates are pinned into the voting application.

[7]Depending on the type of elections, voter's residential address in the electronic population register may determine which candidate list is displayed in the voting application.

[8]The verification application is available for iOS and Android.

decides to verify the vote, the smartphone application asks the voter to scan the QR-code displayed by the voting application. However, it is important to note that to reduce the risk of coercion, the legislation sets a limit[9] for both the number of times the cast vote can be verified and to the time window when the cast vote can be verified [7]. The verification application uses the vote reference (vr) to download the signed and encrypted ballot $Sig(Enc(c_v, r))$ along with the timestamp (ts) issued by the registration service. Next, the timestamp is verified, and the candidate's name (c_v) is displayed. The candidate's name is recovered with the help of the election public key and ElGamal randomness (r) that was read from the QR-code.

Once the election period ends, the vote collector and the registration service digitally sign their corresponding datasets and send them to the processing application. Thus, the processing application receives a set of signed ballots (D_C) from the vote collector and a set of signed timestamps (D_{RS}) from the registration service. The processing application checks that the registration service registered each ballot and that the collector delivered over all of the registered ballots to the processing application. Next, the stored ballots are processed to remove ballots that were cancelled by re-votes or paper votes. Thereafter, the signatures of the remaining ballots are separated. The set of encrypted votes (B_1) is sent through a re-encryption mix-net that re-randomises the encrypted ballots to prevent the ballots from being linked to voters' signatures, and thereby to their identities. The mixing application outputs a proof for the shuffle ($Proof_{mix}$), which can be audited to verify that the mix-net behaved correctly. The proof shows that the set of encrypted ballots B_1 inserted into the mixing application was re-encrypted into the set of ballots B_2 without changing the votes contained in the ballots.

The resulting set of re-encrypted votes B_2 is moved to an air-gapped machine (vote counting server) for decryption and tallying. However, the election private key is not available to that machine during the election period. After key generation, the private key was secret-shared and distributed between a predetermined number of trustees [90, 91]. To perform decryption and tallying, a threshold number of trustees have to use their smart cards, which contain the key shares. Once the private key has been restored, the air-gapped machine decrypts the votes. In addition, it provides a proof of correct decryption ($Proof_{dec}$), which lets auditors to verify that the election results were calculated from the set of re-encrypted votes B_2 that originated from the mixing application. The timestamps from the registration service, proof from the mixing application ($Proof_{mix}$), and proof of correct decryption ($Proof_{dec}$) make it possible for auditors to verify that all ballots were signed by eligible voters, the votes were correctly counted, and no votes were added, modified, or removed.

Configuration of the i-voting system has to be changed for each election. For example, a new key pair is generated, and the corresponding public key is dis-

[9] `https://www.riigiteataja.ee/en/eli/ee/513012020003/consolide/#para48b6`

tributed to the voters with an updated version of the voting application. For the parliamentary elections in 2019, a 3072-bit ElGamal key pair was generated.[10] It is estimated that ElGamal encryption that uses 3072-bit keys can provide confidentiality at least until 2028 [12].

5.1.2. Security properties and trust assumptions

IVXV documentation does not contain an explicit list of security properties that the voting system must fulfil. To give a better overview of IVXV, we describe how the security properties rely on trust assumptions. Although some of the trust assumptions have been described in the scientific literature [165], a broader overview of the trust assumptions is provided in Table 6.

Table 6. Security properties vs trust assumptions in IVXV.

	Voter	**Voting device**	**Voting application**	**Verification device**	**Election website**	**E-population register**	**E-ID used only by voter**	**Election observers/auditors**	**Software used by auditors**	**Vote Collector**	**Registration Service**	**Key generation**	**K of N trustees**
Ballot secrecy	●	●	●	●								●	●
Coercion-resistance	●	●					●	●		●	●		
Confidentiality of partial tally												●	●
Cast-as-intended verification			◌	●									
Recorded-as-cast verification			◌	●			●			◐	◑		
Tally integrity								●	●	◐	◑		
Only eligible voters can vote						●	●	●					
One vote per voter							●	●					
Availability					●					●	●		

◌ = displays verification QR-code ◑ = does not to collude with Vote Collector
● = is trusted ◐ = does not collude with Registration Service

Ballot secrecy. A malicious voting device, verification device, verification application, or voting application would be able to violate ballot secrecy. Similarly, the voter can use the verification application to prove how the vote was cast. However, due to the possibility to re-vote, neither the malicious software nor the voter

[10]The same key length was also used for municipal elections in 2021.

can prove whether the cast vote will reach the tallying phase. The server-side of IVXV processes encrypted ballots and the ballots are anonymised before the tallying phase. Thus, to violate ballot secrecy, the election-specific decryption key would have to leak either during key generation or at least K of N trustees would have to collude to restore the private key.

Coercion-resistance. Re-voting is used to provide coercion-resistance. The voter can also vote on paper, which cancels the i-vote. However, for the mitigation measure to be effective, the voter must be aware and willing to use the possibility to re-vote. In addition, to cast a new i-vote, voters must have control over their e-ID. Furthermore, it is assumed that the coercer does not have control over the voter's voting device as otherwise, it would be possible to detect whether the voter complies or casts a re-vote. It is also assumed that the election organiser does not play the role of a coercer as the Vote Collector and Registration Service can see which voters voted and whether re-votes were cast.

Confidentiality of partial tally. The voting application encrypts the vote with the election-specific public key. Thus, a partial tally can not be found unless the election-specific private key has leaked during key generation or at least K of N trustees would have to collude to restore the private key.

Cast-as-intended verification. The voter can use the verification application to fetch the cast ballot from the Vote Collector. The vote verification application reads the contents of the QR-code displayed by the voting application to initiate vote verification. The verification application opens the downloaded ballot, thereby letting the voter check whether the vote matches the vote that was cast.

Recorded-as-cast verification. The trust assumptions used for cast-as-intended verification are expanded by having to trust that third parties cannot use voter's e-ID to cast a re-vote. In addition, the Vote Collector and Registration Service must not collude as otherwise it would be possible to drop ballots.

Tally integrity. Similarly to recorded-as-cast verification, the Vote Collector and Registration Service must not collude as otherwise it would be possible to drop ballots. To verify tally integrity, election observers/auditors must not be malicious. They have to verify that the set of ballots collected by the Vote Collector matches the set of ballots registered by the Registration Service and that the ballots were properly anonymised and decrypted. Some of these tasks are performed with the help of verification software. For example, the software used to verify mixing proofs and decryption proofs is a trusted component of the voting system.

Only eligible voters can vote. It is assumed that voters' e-IDs are not used by third parties. The election observers/auditors must not be malicious as they have to check that the voting system only allows eligible voters to vote.

One vote per voter. The election observers/auditors have to verify that only the last vote cast by each voter gets tallied. More specifically, it has to be checked that the Processing Application functions correctly. In addition, third parties must not be able to use the e-IDs belonging to other voters.

Availability. The election website should be available as it is used to distribute the voting application. Alternatively, another trusted communication channel between the election organisers and voters could be used to distribute the voting application. To vote, both the Vote Collector and Registration Service have to be available.

5.1.3. Electronic identities used by the Estonian i-voting system

Estonian i-voting system allows votes to be cast from a PC by using an ID card[11] or Mobile-ID. These e-IDs contain cryptographic keys that are bound to an identity via X.509 public-key certificates. While holding an ID card is mandatory for Estonian citizens who are at least 15 years old [3], having a Mobile-ID is voluntary.

There have been multiple generations of ID cards used in Estonia. While the first generations of ID cards used RSA keys, this is no longer the case. After a flaw was found in Infineon's RSA key generation algorithm in 2017, the cards were updated to use elliptic curve cryptography (NIST P-384 curve) [254]. A sample of the ID cards issued since 2018 is depicted in Figure 19.

Figure 19. A sample of an ID card (identity card) issued since 2018 [263].

The ID card is interfaced with a computer (or less commonly to a smartphone) via a smart card reader, and dedicated software is required to communicate with the chip. To unlock the private key operations on the ID card, a valid PIN code has to be entered. However, unless a PIN pad-based smart card reader is used, the PIN codes inserted from a keyboard could be read by malware that has infected the corresponding computer. While such an attack is practically feasible to conduct, by the time this thesis was written in mid-2021, there have not been any public records of malware abusing access to the ID cards [254].

Mobile-ID is an e-ID designed to be used on mobile devices. The cryptographic functionalities of the Mobile-ID are provided by a smart card chip that is distributed to the end-user in the form of a special SIM card.[12] It can be acquired

[11] It is also possible to vote using other ID-card-based electronic identities, for example, digi-ID.

[12] It functions as a regular SIM card but, in addition, contains cryptographic keys and a USIM application that provides the Mobile-ID functionalities [59].

from a mobile operator, but an ID card must be used to activate the corresponding certificates [262]. Similarly to the ID cards, Mobile-ID also relies on elliptic curve cryptography (NIST P-256 curve) with the Mobile-ID SIM card containing separate key pairs for authentication and for issuing digital signatures [59, 25]. However, unlike ID cards, communication is implemented via OTA (Over The Air) SMS messages [59].

5.2. Client-side issues in the Estonian i-voting system

Section 2.7 described the conflicting requirements for voting systems. As a result of the conflicting requirements, i-voting protocols cannot completely fulfil all of the security requirements that are usually expected from voting systems.

The design choices for the architecture and security features of the Estonian i-voting scheme were made at the beginning of the new millennium. They were affected by the available technology and computing resources. Thus, the initial version of the Estonian i-voting system was made coercion-resistant, but neither individual nor universal verifiability was implemented.[13]

The advancement of technology has made it possible to use zero-knowledge proofs, run mix-nets and conduct end-to-end verifiable elections. However, applying these technologies is not straightforward.

By being end-to-end verifiable, the voting protocols provide a receipt for the voter regarding their choice. The report about end-to-end verifiable internet voting by Kiniry *et al.* contains the following claim "No usable E2E-VIV protocol in existing scientific literature has receipt freedom when the voting computer is untrusted." [188]. Thus, in the case of i-voting, end-to-end verifiability may simplify coercion by giving voters means to prove how the vote was cast. The risk of coercion is one of the factors that complicate the decision-making regarding the introduction of end-to-end verifiability.

The security guarantees offered by i-voting systems often depend on the correct implementation of mix-nets and zero-knowledge proofs, making these components a critical part of the voting system. The importance of these components was illustrated in 2019 by the findings of severe implementation faults in zero-knowledge proofs and mix-nets, which were used in the Swiss i-voting system developed by Scytl [154, 104]. This example once again supports the requirement to open-source the software components used for i-voting along with the supporting documentation.

In addition to the advancement of technology, the capabilities of attackers have also changed. The internet provides access to study materials and tools for people who take the criminal path. Furthermore, it is now common for governments to buy exploits and provide them for their cyber attack units [13]. Thus, the threat

[13] Individual verifiability was implemented in 2013 [169], and the server-side was made verifiable in 2017 [165].

landscape is not the same as it was back in 2005 when the Estonian i-voting system was piloted. As the threats and risks can change over time, the security requirements and the existing protective measures used in voting systems have to be regularly re-evaluated. In an ideal world, a fresh risk analysis would be made before each election to find an optimal security configuration given the existing threat level. That could be one of the inputs for deciding the balance between coercion-resistance and verifiability.

Some of the weaknesses in the Estonian i-voting system are inherent to the protocol's design. As a by-product of providing coercion-resistance via re-voting, the guarantees offered by individual verifiability are weakened. While the voter can verify that the vote was correctly received and registered, it does not guarantee that it is tallied. By using individual verification, the voter cannot detect whether malware has cast or will cast a re-vote. Such an attack can endanger vote integrity even when the server-side is auditable and works as intended.

Other general issues, not specific to the Estonian i-voting system, also affect vote privacy. For example, when the voting protocol does not use code voting to hide the names of the candidates, the voting device sees how the voter voted. This can be a problem if the device is infected by malware or monitored by third parties. In addition, information used for verification has to remain private as it also could reveal how the vote was cast.

5.2.1. Malicious access to voter's electronic identity

One of the weak spots in the Estonian i-voting system lies in the voter's computer. Malware on the voter's device could attack both vote secrecy and vote integrity. However, having control over the voter's computer while the vote is being cast may not be sufficient if an attacker aims to change the vote without being detected by vote verification. The real risk lies in the possibility of an attacker having access to the electronic identities belonging to the voters. That would give an attacker the possibility to cast votes on behalf of voters without being detected by the vote verification application.

In general, there are two ways how an attacker could access voters' electronic identities. First, an attacker could compromise the device interfaced with the e-ID. Second, a systematic failure related to the e-IDs could lead to an attacker accessing the corresponding cryptographic keys. The former can happen when the infected computer forwards attacker's requests to the e-ID. The latter could happen via novel cryptographic attacks, mismanagement of the issuing process of e-IDs, and failure of the hardware used to store the cryptographic keys. While there have been multiple security issues with the Estonian ID cards [253], they are not known to have affected i-voting.

When comparing the ID card and Mobile-ID, one of the differences lies in the fact that the Mobile-ID SIM card is always connected to the device while the ID card may be detached or interfaced with a computer via a PIN pad-based reader.

The possibility to use a PIN pad-based reader has multiple advantages. First, it prevents keyloggers from accessing the PIN codes. Second, some card readers contain a PIN-firewall, which only allows PIN codes to be entered by pressing the buttons on the reader. This is an efficient measure against malware as it requires human interaction before the ID card can be used. Thereby, a PIN firewall effectively prevents malware from being able to authorise private key operations. Although the card readers that have a PIN firewall significantly increase the security level, they are rarely used, and the devices have mostly disappeared from the local market.

Although an ID card could also be interfaced with a smartphone, it is uncommon due to the lack of suitable card readers and compatible web browsers. This leaves Mobile-ID as the primary target when the attacker aims to use infected smartphones to influence the election outcome.

Now, let us view a few options on how malware on a voter's device could attack vote integrity by having access to a voter's e-ID. First, it is possible for malware to silently interfere with the voting process to change the vote while relying on the voter to use the e-ID to submit the modified vote. Second, malware may capture the PIN codes and wait until an ID card is connected to the computer to cast a vote in a background process [293]. A similar attack could be launched on smartphones if malware is able to confirm Mobile-ID transactions. Third, if malware can not access the PIN codes, it can wait until the victim uses the ID card for everyday purposes like online banking, which offers a possibility to trick the user into entering PIN codes and thereby unknowingly signing the ballot. Fourth, suppose the attacker only wants to interfere with the way how the official voting application functions. In this case, it is possible to trick the voter into voting twice such that the verification information meant for the first voting session is reused in the second voting session [258]. Such an attack would allow an attacker to change the vote that is cast in the second voting session while providing the voter with valid vote verification information about the vote cast in the first voting session.

The first option is not optimal for an attacker as the voter might use individual verification to check whether the voting system correctly received the vote. The success of the third and fourth attack depends on the victim's behaviour that is difficult to predict, and may thus result in the attack being detected.[14] Therefore, the best option for an attacker is to take the second path, that is, to access PIN codes along with the targeted e-ID.

In case an attacker aims to use an ID card maliciously, the attack scenario would be straightforward. The critical step of the attack lies in distributing and using malware without being detected. Once the distribution hurdle is passed,

[14] In both cases, the victim notices anomalies, which may get investigated or reported. For example, suspicion may arise if the victim fails to complete a bank transaction after entering PIN2. If the incident gets reported to a bank or the state, it is possible to identify which service requested a timestamp for the signature. Similarly, if the voter fails to vote on the first try, the incident may get reported and investigated.

malware can try to capture PIN codes, detect voting events, and cast votes. Thus, one option for an attacker is to integrate an independent voting client into malware that would idle until the ID card becomes interfaced with the computer. Another option is to forward the access to the smart card reader over the internet, which would make it possible to use the official voting application on the attacker's computer to cast a vote.

When Mobile-ID is targeted, there are three possibilities for an attacker to cast a vote. First, an attacker could attempt to phish the voter into entering the PIN codes and thereby unknowingly cast a vote. However, the Mobile-ID messages display the name of the service requesting authentication or signature, thereby making it easy to detect such attacks. Second, an attacker could initiate the voting process himself and use a communication channel to request malware running on the victim's smartphone to authenticate the voter and sign the vote. Third, an attacker could integrate an independent voting client into malware so that malware could vote directly from the smartphone.

However, infecting the voter's PC gives more information to the attacker as it allows an attacker to monitor the voter. By having such knowledge, a re-vote can be cast after the voter has voted to lower the probability of the malicious vote being detected or accidentally overwritten by a re-vote of an honest voter.[15] The voter would not detect such an attack as the Estonian i-voting system does not provide end-to-end verifiability. In addition, there is no feedback channel to notify the voter of a given vote. However, a large scale re-voting attack could be detected by analysing the server-side logs to identify anomalies [166].

5.2.2. Freshness of vote verification

The design of individual vote verification used by the Estonian i-voting system follows the general principle that a voting system should be coercion-resistant. According to this principle, the system's design has to limit the possibility of an attacker accessing the verification result. Therefore, to prevent the voter from vote-selling and being able to prove to a coercer how the vote was cast, the Estonian i-voting system lets a voter verify the vote up to three times during a limited time window, which has historically been between half an hour and an hour.[16]

The vote verification protocol was not designed to check vote freshness, which means voters are not notified whether the vote they are verifying is still valid. Such a design was supposed to provide additional coercion-resistance by preventing coercers from identifying re-votes. As a downside, if the vote verification

[15]The voting application displays a notification if the authenticated voter has already cast a vote. Therefore, malware could initiate a new voting session and decide whether to proceed or cancel the voting session based on whether a vote has already been cast with the credentials available to the malware. However, the cancelled sessions leave a trace to the logs that are stored on the server-side of the voting system.

[16]Riigikogu Election Act states that vote can be verified for a limited number of times during a limited time window [7].

application does not check vote freshness, it can not guarantee that the verified vote has not already been invalidated by a re-vote. However, as of 2021, IVXV can be configured to check the freshness of the vote.[17] Whether the functionality will be enabled depends on the election organisers.

Individual vote verification was added to the Estonian i-voting system to detect attacks that prevent the vote from reaching the voting system and attacks that replace the candidate number during the vote casting process. However, the design of the vote verification system leaves several options for performing an attack on the client-side, which can not be detected by verifying the vote.

It was already described in Section 5.2.1 that malware having access to a voter's e-ID can re-vote and thereby cancel the original vote. Such an attack can happen instantly after the voter has voted. If the verification system does not check for vote freshness, a voter can successfully verify a vote that has already been invalidated by a re-vote [161]. Thus, voters may falsely believe that the vote is guaranteed to be tallied after it has been successfully verified. However, even if vote freshness is checked, malware could cast a re-vote after the vote verification period has expired. One way to detect such attacks is to analyse server-side logs to identify anomalies in the re-voting pattern [166]. Whether and how such information could be used is up to the lawyers and the election organiser to decide.

As the lack of a freshness check has not been explicitly mentioned in the voting application, it would be interesting to conduct a user study to determine how many voters have the wrong perception about integrity guarantees provided by the vote verification application.

5.2.3. Suppression of vote verification

As malware can influence the voting process, it can either drop or change the vote. To deter and detect such malware, vote verification can be used, which allows the voter to detect malicious intervention by checking whether the vote reached the vote collection server. However, malware can try to outsmart the verification system by first intervening in the voting process and then preventing the voter from verifying the vote by leaving an impression of a software glitch.

In an ideal scenario, an attacker would be able to predict which voters will not verify their votes and, based on that, selectively apply the attack. However, when such prediction is not feasible, an attacker could crash the application after the verification code is briefly shown on the screen or, as an alternative, slightly modify the displayed QR-code to make it unreadable by the verification application. To better understand the attack, let us discuss the case when the voter wants to verify the vote and the case when the voter ignores vote verification.

In case the target of the malware is a voter who does not verify the vote, the attack remains undetected by the voter and is not reported to the election organ-

[17] It was not yet possible to check vote freshness in the local elections held in 2021.

iser.[18]

If the voter was planning to verify the vote, an attack that prevents the vote from being verified could lead to multiple possible outcomes. First, the voter may ignore the crash or problems with displaying or reading the QR-code, which leaves the attack undetected. Second, if vote verification fails, the voter may try to re-vote, which may result in the attack being undetected in case the vote verification succeeds. Third, the voter may inform the election organisers of not being able to use vote verification, which may lead to detecting the attack.

Malware can reduce the probability of it being detected by only corrupting the voter's first voting attempt and remaining passive if the voter re-votes. The probability of the aforementioned attack staying unnoticed depends on the scale of the attack and the behaviour of the voters. The historical evidence in Estonia shows that the percentage of voters who verify their votes ranges between 4–5% [88, 161]. However, only some of these voters report the issues to the election organiser, as the voter must do the reporting manually.[19] Interestingly, in case of verification errors, The Estonian National Electoral Committee suggested first to re-vote and, if this does not resolve the issue, recast the vote on paper and notify the election organiser [169, 86]. However, for some types of errors, the vote verification application asks the voter to notify the election organiser (see Figure 20).

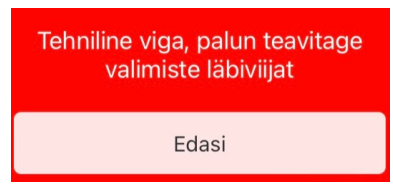


Figure 20. The screenshot shows an error message displayed by the vote verification application. The message says that there is a technical error and asks to notify the election organiser. For example, such an error message that asks the voter to notify the election organiser can be triggered if the verification application is unable to read the QR-code. The screenshot was captured during the local elections in 2021.

5.2.4. Limited means for dispute resolution

By providing the voters with the means to verify their vote, a system is needed to manage the issues detected by the voters. However, it was stated already in 2014 that one of the problems with the Estonian i-voting scheme is the lack of a clear dispute resolution protocol [169]. For example, it is unclear how to distinguish valid claims about failed vote verifications from malicious ones, aiming to

[18] Malware might be identified by honeypots or endpoint security software on the voters' devices.

[19] In the European Parliament Elections held in 2014, the iOS-based vote verification application contained a bug which prevented some voters from verifying their votes. However, out of the estimated 42 voters impacted by the bug, only four reported it to the National Electoral Committee [167].

discredit the voting system.

The lack of a proper error handling system and a dispute resolution protocol can result in two negative outcomes. First, malicious voters could falsely claim that verification failed and thereby question the election integrity. This has not happened so far. Second, the suggestion to re-vote in case of a single verification error provides an attacker with a way to escape detection. For example, an informed attacker can interfere only when the first vote is cast and stay passive once the voter tries to re-vote. Thus, the suggestion to notify the election organiser only after failed re-vote verification[20] should be re-evaluated [86].

5.2.5. Privacy leak due to the untrusted voting or verification device

Besides endangering vote integrity, malware on an end-user device can violate vote privacy in two different ways.

First, by having control over the voting device, malware can take screenshots or screencasts of the voting process. The impact of such an attack depends on the attacker's goal. The success of a coercion attack is questionable if the attacker threatens to publish the screenshots or screencasts as they do not provide strong proof of how the vote was cast. However, the situation gets dire when the coercer uses malware to monitor whether the coerced voters comply. When the monitoring is done without voter's knowledge, the information regarding a re-vote can leak, which could later be used for persecution.

Second, malware can violate vote privacy by copying the verification information displayed in the QR-code. The vote verification QR-code contains a reference to the cast ballot and a random value used for ElGamal encryption of the vote. Thus, malware could use the information to download the signed and encrypted ballot from the voting system. However, if the voter uses up all three verification queries, such an attack fails or is detected. Even when the verification API can not be used to query the signed ballot, the same information could be found from the voting application's memory. By having access to the signed and encrypted ballot along with the ElGamal randomness, it is possible to provide strong proof that the voter cast the corresponding vote at the time when the signature was issued.[21]

Although the QR-code is available locally on the voting device, the verification information can also be captured by a smartphone, which is used to verify the vote. Thus, the contents of the QR-code can leak both through an infected voting device and an infected verification device. However, the possibility to re-vote offers partial mitigation as voters can claim that the information from the QR-code represented an initial vote that was replaced by a re-vote. The mitigation is

[20] There is conflicting information regarding what should be done in case verification fails. The verification application asks the voter to contact the election organiser but the instructions for vote verification suggest the voter to re-vote first [86].

[21] The registration service adds the timestamp after the ballot has been sent to the vote collector. As the timestamp is returned to the voting application, it could be used as a cryptographic proof to determine when the vote was cast.

only partial, as the information could be used to discredit these voters, especially if the targeted voters hold apolitical positions.

The privacy issues related to voters' devices are not unique to the Estonian i-voting system as there are very limited options for assuring the vote is not leaked on the client-side of the voting system.

5.3. Possible improvements to the client-side of the Estonian i-voting system

The mitigation measures for an i-voting system can be classified into three categories: prevention, detection, and recovery. Out of these, the detection measures are critical when the aim is to deter an attacker from large scale election manipulation. Thus, an i-voting system has to have means to detect manipulation of vote integrity both on the voter's side and on the server-side. However, detection measures on their own are not sufficient to secure elections as deterrence may not prevent an attack. For example, if attacking vote integrity is sufficiently easy on the client-side, the attacker effectively has the means to raise the question of whether i-voting should be postponed or cancelled and thereby affect elections. Thus, the voting system should aim not only to detect attacks but also make them very difficult to execute. Thereby, the prevention measures aim to reduce the number of capable threat actors and the scalability of the attacks.

While it is possible to lower the probability of attacks, eliminating every single one is not realistic. Thus, the election system also has to contain recovery measures, which can be activated if an attack or system failure is detected. For example, the Estonian legislature makes it possible to postpone or cancel i-voting if a large scale attack is detected, which could affect the outcome of the elections [7].

Most of the improvements proposed in the following sections, along with the security analysis, originate from a feasibility study regarding mobile voting [32] and from our research papers [161, 120, 162].

5.3.1. Introduction of a feedback channel

One way to mitigate the misuse of e-IDs is to create a feedback channel to notify the voters of votes given on their behalf [32, 161]. It would work as a detection measure that could be triggered every time a ballot is received by the Vote Collector server and registered by the Registration Service.[22]

However, the feedback channel must not reveal information about the contents of the vote to protect vote privacy. The notification system would make it possible to detect when a vote is cast via unauthorised access to the voter's e-ID. By getting a notification, voters would realise that someone with either physical or logical

[22]The Vote Collector and Registration Service are operated by different organisations. According to the trust assumptions stated in Section 5.1.2, the Vote Collector and Registration Service are trusted not to collude.

access to the e-ID has used it to cast a vote. Thus, the measure would help detect malware that cast votes on behalf of the voter. Furthermore, it would also allow eligible voters to detect if their e-ID has been misused to vote even when they did not intend to participate in the elections or are not using the i-voting system to vote.

While the feedback channel can increase the integrity guarantees by allowing to detect malicious re-votes, it also impacts coercion-resistance. However, to analyse the coercion-resistance of a feedback channel, the details regarding the implementation become relevant.

The cornerstone of coercion-resistance in the Estonian i-voting scheme is the possibility to re-vote after being coerced. For the measure to be effective, the coercer should not be able to detect whether a re-vote was cast. This has direct implications for the design of the feedback channel. While the notification channel must be easily accessible for all eligible voters, it also has to be somewhat covert to prevent the coercer from finding out whether a re-vote was cast. In addition, malware must not be able to invisibly manipulate or turn off the notifications.

The notification must be timely so that the voter would have sufficient time to re-vote after detecting the misuse of the e-ID.[23] Thus, it becomes clear that the postal notification channel is not suitable for voters who vote from abroad as the notification would not reach the voter in time. Out of the digital channels, two cover a sufficiently large part of the electorate – the email channel and the SMS channel. Unfortunately, neither of these channels provides end-to-end encryption nor anonymity. Therefore, by monitoring the feedback channel, it may be possible to identify the voters who cast an i-vote.[24]

The notification channel must be mandatory for it to be effective as otherwise, an attacker could turn off the notifications. A slightly weaker alternative could be based on an opt-out system that relies on an independent communication channel to inform the voter if the notification functionality has been turned off.

One of the weaknesses of the feedback channel is the possibility of malware intercepting and deleting the notifications. In general, there are three main ways of mitigating such risks. The first option is to create a new e-service for storing the notifications, which would provide the voters with read-only access to the notifications for a fixed period of time. The second option is to require the voters to use an independent authentication method or multiple authentication methods to access information about the notification. The third option is to use multiple notification channels to make it more difficult for malware to interfere.

The introduction of the feedback channel to the election context could be eased by making it available for all digital signatures given with state-issued e-IDs. This

[23] An interesting situation appears if an eligible voter who does not want to participate in the elections detects misuse of the e-ID. In that case, the suggestion to re-vote does not help alleviate the situation.

[24] In principle, it is also possible to use the feedback channel to notify the voters who cast a paper vote. In that case, monitoring the feedback channel would reveal who participated in the elections.

would also make it possible to create a generic notification about the usage of the signing certificate. Thus, a coercer would have to distinguish the voting event from a bank transaction or any other event which requires a digital signature to be issued. However, introducing a feedback system for the whole Estonian e-ID system is not straightforward as it would affect the everyday processes of most businesses and individuals. In addition, overflowing the users with information could result in the users turning off the notifications.

Further studies have to be conducted before the feedback channel can be implemented. For example, it should be analysed how to guarantee that the voter receives the notification. In addition, it has to be evaluated how the new functionality would affect the coercion-resistance of the Estonian i-voting system. The latter was studied by Koitmäe *et al.* in a paper published in 2021 [191].

5.3.2. Educating the voters

For re-voting to function as an efficient anti-coercion measure, voters have to be aware of the possibility. The same also holds for individual vote verification. Unless voters are aware of the risks, they may not bother to verify their votes, which is also evident by the low verification rate among the cast votes [88, 161]. Furthermore, by being aware of the risks related to the inability to verify the vote, it is likely that more voters would report the software glitches, which in turn would help to detect attacks that suppress vote verification.

Similarly, unless voters acknowledge the risks posed by the end-user devices, there is no incentive to use smart card readers that contain a PIN firewall. While there is no demand for such smart card readers, the market has no reason to offer them to consumers. In addition to the difficulty of finding compatible PIN pad-based smart card readers, it is even more difficult to verify whether the reader has the advertised functionalities. This is illustrated by an example from 2011 when a smart card reader integrated into a keyboard was falsely believed to use a PIN pad [251, 254]. Thus, secure PIN pad-based smart card readers should be specially labelled and promoted by the state's election organisers. However, voters are unlikely to acquire such devices unless they are aware of the risk that, in principle, malware could access their ID cards.

In general, to increase the trust in the election system, voters who wish to understand how the system works should be able to access up to date information about the inner workings of the voting system. Besides that, it should be possible to find answers to the questions regarding the risks of both i-voting and traditional paper voting. Therefore, the whole i-voting system should be well documented, and the trade-offs made in the design choices explained, such that interested parties could access the information. This would effectively serve as a tool to counter false claims regarding the i-voting system. Unfortunately, this is not the case as there is a lack of such documentation and educational materials.

5.3.3. Requirement of two signatures to confirm the vote

The individual verification solution used by the Estonian i-voting system relies on the assumption that attackers do not simultaneously control both the voter's PC and smartphone. A similar assumption could be used to make it more difficult for malware to cast a vote.

Currently, the voter can choose whether to sign the ballot with an ID card on a PC or with a Mobile-ID on a smartphone. This could be changed by introducing a prevention measure requiring signatures from two different e-IDs to cast a vote. One of the e-IDs could be the ID card, and the other one a smartphone-based e-ID. This would distribute the vote confirmation process between two physical devices.

While the ID card can be interfaced with a smartphone, this possibility is rarely used due to the lack of suitable card readers. There is little incentive for buying an additional card reader when other e-IDs are easier to use on a smartphone. In addition, some of the ID card functionalities are not provided on smartphones. For example, many mobile browsers do not support extensions, preventing ID card-based authentication from functioning in the browser.

However, the new ID cards have near-field communication (NFC) interface [35], which may lower the barrier of using the ID card on smartphones. Regardless, while the official voting client is supported only on a PC, the requirement of having two signatures would force an attacker to compromise two devices to perform the attack.

While the proposed requirement to use two signatures would make it more difficult to conduct and scale attacks, it would also reduce the accessibility of i-voting as not all voters have access to a smartphone or to a second state-supported e-ID.[25] However, by decreasing the number of i-voters, the possible impact to the election result from a successful malware attack on voters' devices is reduced, which on its own acts as a prevention measure. Thus, it has to be decided whether requiring two signatures to cast a vote and thereby lowering the number of i-voters is a proportional measure.

5.3.4. Freshness check of the ballot

The individual verification that has been used up to 2021 does not check the freshness of the ballot that is being verified. The aim to provide coercion-resistance during vote verification makes it possible for malware to cast an instant re-vote without the voter detecting it via vote verification. Thus, voters can not prevent a re-vote attack by disconnecting an e-ID for the rest of the election period once the vote has been cast and verified.

However, the individual verification protocol could be slightly extended to check for the freshness of the ballot.[26] Thereby it is possible to provide the voter

[25] As of 2021, Smart-ID, which is widely used, but not issued by the state, can not be used to cast a vote.

[26] The issues with the lack of a freshness check were highlighted by the author of this thesis. This

with a guarantee that the vote was not instantly overwritten by a malicious re-vote. Similarly, it would be possible to detect attacks that try to replay verification info from previous voting sessions [258]. To provide the freshness guarantee, the verification application must only verify votes that have not been re-voted. If the back-end of the voting system detects that an invalidated vote is being verified, the voter must be provided with a clear error message, which states that the vote has been invalidated by a re-vote. Thereby, the freshness check would act both as a detection measure and a deterrence measure.

As verification can only be performed in a limited time window, it can not guarantee that malware will not re-vote later. However, a voter that is aware of the risks could either remove the ID card from the reader for the duration of the i-voting period or vote at the end of the i-voting period to ensure that a re-vote could be detected by vote verification. In this case, the voter should be able to verify the vote after the i-voting period has ended, which would give a strong guarantee that the original vote was not substituted by malware.

The freshness check would not have a significant negative effect on coercion-resistance. If the original vote is cast under coercion, the voter can re-vote after the brief verification time window has passed. However, if the voter is coerced at the very end of the i-voting period, it is likely that the coercer is present and tries to prevent the voter from re-voting. In addition, the voter still has the possibility to go to the polling station to vote on paper. Starting from October 2021, i-voters can cast a paper ballot also on the election day and thereby cancel the previously given i-votes [10, 7]. Therefore, introducing the freshness check does not significantly weaken a voter's ability to re-vote if faced with coercion.

5.3.5. Zero-knowledge proof about vote validity

The official i-voting application does not provide an option to cast invalid votes. However, there are no restrictions on the protocol level that would prevent a voter from filling the ballot with a value that does not match any of the candidates [165, 195]. To do that, the voter would need to either modify how the official voting client behaves or create an unofficial voting application.

Such an issue was highlighted in 2011 when an invalid vote was detected during the tabulation phase of the Riigikogu Elections [164]. Similarly, one invalid vote was seen in 2015 [166, 303].

Invalid votes can give rise to bigger issues if they become more frequent. For example, if invalid votes become frequent, voters may start to question whether the voting application is functioning correctly. In 2011 it was decided that an invalid vote would not be investigated further to prevent vote secrecy from being violated [164].

---

led the way for a fix to be implemented in IVXV. As of 2021, IVXV can be configured to check the freshness of the vote. However, this functionality was not enabled during the municipal elections in October 2021. Whether the functionality will be used depends on the election organisers as they have to decide how to balance coercion-resistance with the possibility of detecting re-voting attacks.

As some voters wish to reflect their political preferences by casting invalid or blank votes, this option could be provided in the official voting application. However, it could open up a way for coercers to force voters to abstain from elections [195]. In practice, such an attack would be effective only if the invalid vote would leak in the tallying phase.

Therefore, it is recommended to study whether zero-knowledge proofs could be added to IVXV to prevent invalid votes from reaching the tabulation phase. For example, the voting client could add a zero-knowledge proof about set membership to make it possible for the vote collector or processing application to verify that a ballot contains a valid vote.

5.3.6. Voting with special hardware

The official voting application for the Estonian i-voting system only works on desktop platforms – Windows, macOS, and some Linux distributions [87]. As it can not be guaranteed that these platforms do not have issues with malware or that the voters themselves do not tamper with the application, it is not possible to consider voting software running on one of these operating systems as a trusted component.

One may claim that antivirus software can be used to counter the threat of malware. However, the protection offered by antivirus products is not guaranteed to detect every single threat. First, antivirus products are optimised for detecting known malware and do not work as well for unknown threats [297]. Second, an advanced attacker can test different antivirus products to see whether they detect malware built on top of a fresh exploit. Third, an average voter is unlikely to use an advanced endpoint security solution.

However, it turns out that in order to detect a large scale attack, only some voters have to detect the malware. In the context of i-voting, it is sufficient to identify a malware campaign targeted against i-voting, to call off or to postpone the i-voting event. Thus, given a large number of voters along with a diverse set of antiviruses and endpoint security systems, it is difficult for such malware to remain completely undetected.

The next logical question to ask is how much damage a targeted malware campaign can do. The answer to this question depends on the percentage of affected voters, whether the attack is detected and how it is mitigated. Thus, it has to be evaluated what is the critical number of manipulated votes that could change the outcome of the elections. However, even when that number is available, it may be difficult to estimate the spread of malware among the voters, given that a malware sample is detected. Therefore, selecting a threshold for postponing or cancelling the i-voting for the given election becomes critical in deterring an attack. Unfortunately, finding out whether an attack could have changed the election outcome can only be determined after counting the votes.

Some of these aforementioned issues can be mitigated by introducing addi-

tional safeguards to the voting system. For example, malware attacks could be deterred and detected by introducing end-to-end verifiability, providing a feedback channel, or requiring multiple signatures to cast votes.

However, the aforementioned mitigation measures would not solve privacy issues given the infection of the voting device. There are two main ways how to prevent malware from violating vote privacy. First, a code-based voting protocol could be introduced to hide the candidate identities from malware. However, this would require a complete overhaul of the existing voting protocol while also reducing the usability of the voting system. Second, the voting device could be made secure against malware. Unfortunately, it is not feasible to secure the PC of an average voter.

However, there is also a third option. The issues with malware could be partially mitigated by introducing a personal single-purpose voting device, which could only be used for voting. The following sections expand this idea by describing how a microcontroller-based voting device can be created for the Estonian i-voting system and thereby reduce the risk of vote privacy being violated.

5.4. Building an independent voting device for the Estonian i-voting system

The back-end components of the Estonian i-voting system, along with the verification application are open-source since 2013 [159, 58]. However, the source code of the official voting application has never been published. The decision to not open-source the voting client has raised questions. For example, OSCE/ODIHR missions have criticised the lack of transparency [293, 127].

The source code of the client application has not been published due to the belief that thereby some attacks can be prevented. For example, by relying on an open-source client application, it would be easy to create fake voting applications similarly to a proof-of-concept malware demonstrated in 2011 [164, 239]. In addition, it is claimed that the publication of the client application's source code would reveal the defensive measures used by the application [164].

While obfuscation-based mitigation measures built into the voting application may have been a deterrent in the past, its usefulness nowadays is questionable, especially as malware does not have to use the official voting application to cast a vote. In addition, the threat agents who target elections with technical means have most likely access to advanced reverse engineering capabilities. Such skills have become more accessible over the years due to the publication of tutorials and tools.[27]

Thus, obfuscation and relying on closed source software does not guarantee that the functionalities of the software product remain hidden. This was illustrated already in 2014 when security researchers were able to reverse engineer

[27] For example, the release of Ghidra in 2019 by the NSA gave free access to reverse engineering tools to a wider community [112]

parts of the Estonian i-voting client [293]. A more recent example originates from 2020 when researchers were able to reverse engineer the Voatz voting application regardless of the obfuscation methods that attempted to hide its functionalities [291]. As a result of the research, multiple security issues were found in the Voatz voting system. This further highlights the need to publish the source code so that the community of researchers can review and audit the security-critical software components.

As the documentation for the Estonian i-voting protocol along with the API description is public, it is possible to write an independent voting client. While this has been claimed to be possible, such voting clients had not been open-sourced for the public before 2021. Thus, we initiated a project to create the first open-source implementation of the voting client for the Estonian i-voting system. The main aim of the project was to reduce the attack surface, thereby increasing vote privacy. As a side-product, the proof-of-concept voting client also increases the transparency of the voting system by showing that, in principle, the voters do not have to rely on the official voting application [120].

If an independent voting application would run on a general-purpose desktop computer, it would face the same threats as the official voting client. Thus, it would be affected by malware that could attack both integrity and privacy of the ballot. However, these threats can be significantly reduced by replacing the general-purpose PC platform with a single-purpose microcontroller that only runs the voting application and the required firmware.

A voting client built on top of dedicated hardware would eliminate the threats affecting commonly used operating systems. In addition, the voter would have fewer means of accidentally infecting the voting device as it would not be usable for other tasks. This would result in a significantly smaller attack surface. As a by-product, the significance of bugs in the voting client would be reduced as it would become difficult for an attacker to exploit them due to the limited number of attack vectors.

This would partially solve the issue of having to trust the code quality of an unofficial voting client, which the election organisers have not tested. In addition, the voter may also have less trust in the functionalities of the unofficial voting client. One way to partially solve this issue is to review the source code. However, only a small fraction of voters are capable of that. An alternative is to ask the community to test and audit the source code to improve its quality and ensure that it does not leak the vote. Ballot integrity could still be checked with the individual vote verification application. Nevertheless, the voter would have to assemble the device and install the software to get the desired security guarantees. Thus, an unofficial microcontroller-based voting client is unlikely to be widely used.

It is important to understand that while a microcontroller-based voting client can prevent malware from interfering with the voting process, it can not prevent malware from re-voting. This problem can not be solved solely by creating an independent voting device, as the root of the issue lies in the possibility of malware

having access to voter's e-IDs.

In case either the ID card or the Mobile-ID are used for other activities besides voting, they might be accessed by malware. The voter can, in principle, avoid using the ID card by keeping it detached from the card reader during the i-voting period, but such prevention measure is not available for Mobile-ID. The SIM card of the Mobile-ID contains the signing key but also provides the calling functionality, which makes it inconvenient to remove the SIM. Unless the Mobile-ID SIM is removed and a different SIM is used for making phone calls, the only guaranteed way of preventing malware on an infected smartphone from using the Mobile-ID signing key or authentication key is to suspend the corresponding certificate.

In a way, a microcontroller-based voting client can be compared to a personal voting machine. Thus, before describing our proof-of-concept voting device, we give a brief overview of voting machines.

5.4.1. A brief overview of voting machines

Voting machines were introduced in the United States at the end of the 19th century and the beginning of the 20th century as a part of election reforms. The initiative for election reforms was caused by widespread fraud, which involved vote-buying and ballot box stuffing [178, 157]. Along with the introduction of the Australian ballot, voting machines were thought to reduce fraud.

The first type of voting machine widely used in elections was based on mechanical levers. This kind of a machine was first used in Lockport, New York, in 1892 [157]. By the middle of the 20th-century, lever voting machines had become the standard voting technology in the US [177].

The next technological iteration of voting machines came in the 1960s with the invention of punched cards systems and optical mark-sense scanners that allowed ballots to be computer-tabulated [177]. However, the first direct-recording electronic voting machine was patented by McKay *et al.* in 1974 [219] and used in elections in 1975 [177]. Such machines became known as Direct Recording Electronic (DRE) voting machines. They entered the market and became widespread in the 1980s and by now have become the default voting technology in multiple countries [29, 156].

However, DRE machines have had many security issues that can be linked to the lack of documentation, lack of sufficient protective measures and questionable design choices [111, 40, 27, 37, 28, 320]. For example, researchers have reverse engineered and modified the software running in the voting machines to show that the election result reported by the DRE machines can be tampered with [190, 142, 121].

Some of the security issues with these machines are likely to be caused by the business model used by the companies that sell these machines. Unless there are legal or contractual requirements, vendors do not have an incentive to be transparent regarding the design of the voting machines. Thus, if clients do not strongly

demand transparency and independent audits, vendors can rely on the claim of having business secrets, preventing the technical details from being published. Therefore, it is difficult for auditors and researchers to evaluate DRE machines' security.

One way to mitigate the issues with these machines is to require the devices to produce receipts, which can be used for auditing. In addition, end-to-end verifiability could be integrated into these devices with the help of ElectionGuard [220].

Alternatively, some of the issues related to having to trust the voting devices can be solved by giving the voter more control. There have been projects that attempt to achieve that goal by designing personal voting machines. For example, in 2009, a voting machine design based on Xilinx Spartan-3E 500 FPGA platform was given by Öksüzoglu and Wallach [247]. However, the corresponding hardware was not cheap enough for large scale production as the price for the platform was around $150. In addition, the source code for that project was not published.

5.4.2. Design and implementation of an independent voting client

We decided to build an independent voting client for the Estonian i-voting system on top of a widely used microcontroller platform ESP32, which is a successor of the popular ESP8266 and is also developed by Espressif [300].

That microcontroller was chosen for the voting client by combining the features and the price tag. ESP32 features a dual-core Tensilica Xtensa LX6 microprocessor that operates at 160 MHz or 240 MHz, has 520 KiB SRAM and supports WiFi and Bluetooth v4.2 [302]. The processing power makes it suitable for computing exponentiations for the 3072-bit ElGamal encryption, which is the main bottleneck for creating a microcontroller-based voting client.

The main challenge in the project was to create a voting device that provides all the required functionalities while being usable by the voters. To simplify the development, we decided to only support Mobile-ID for authenticating the voter and signing the ballot. To provide intuitive controls to the voter, we created two different hardware configurations.

The first configuration (basic setup) is composed of a rotary encoder KY-040 and ESP32 that has a built-in 0.96 inch OLED display. The screen has a resolution of 128×64 and is connected to the ESP32 via Inter-Integrated Circuit (I2C). The device is shown in Figure 21. We used the following boards for testing this setup: Wemos Lolin32 OLED and Heltec ESP32 Wifi Kit. These boards come with 4 MB of integrated flash memory. The built-in screen is sufficient for displaying the list of candidates, and the rotary encoder provides the means to select the candidate from that list. The schematics of the layout can be found in our paper that gives a more detailed overview of how the device was built [120].

In addition to the previously mentioned hardware, a breadboard and jumper cables are needed to assemble the device. While building the voting device, it is important to know the pin layout of the corresponding board to interface the

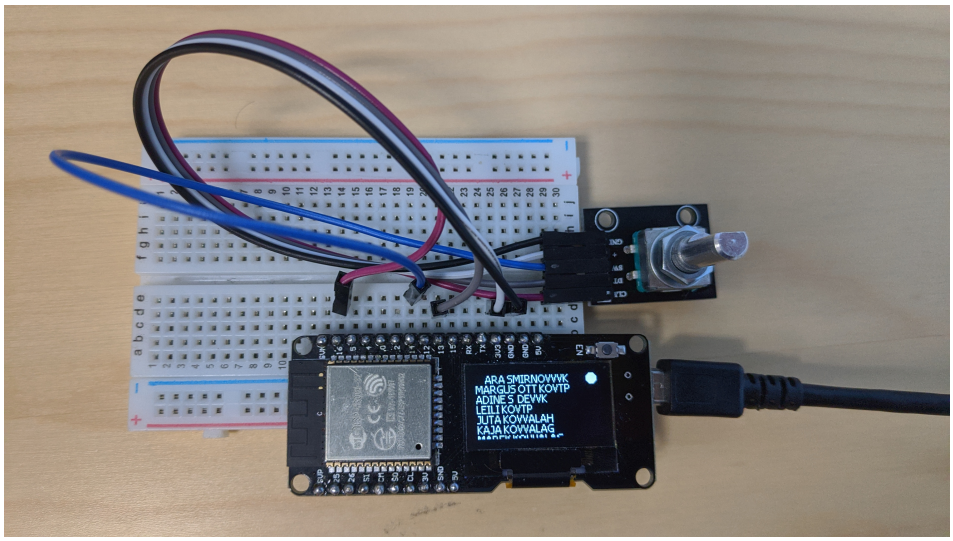


Figure 21. The displayed voting device is built on top of Wemos Lolin32 OLED and uses a rotary encoder to get user input. The image originates from [120].

components correctly. In the basic setup, such information is required to connect the rotary encoder and to configure the software to use the screen. The pin layout for the tested boards is given in Table 7.

Table 7. The pin layout for Wemos Lolin32 OLED and Heltec ESP32 Wifi Kit [120].

	OLED pins			KY-040 pins		
	SDA	SCL	RST	CLK	DT	SW
Wemos Lolin32 OLED (40 MHz)	5	4	-	16	13	14
Heltec ESP32 Wifi Kit (26 MHz)	14	13	16	19	21	23

The downside of the basic setup is its small screen size, which does not fit the QR-code used for vote verification. As verification is a central piece of the voting system, the functionality can not be omitted. Thus, as an alternative, the official smartphone-based verification application was extended to use a Bluetooth channel for reading the verification info.

The voter would have to pair the voting device with the verification application by entering a PIN code for Bluetooth pairing in order to create an encrypted channel for the transmission of the verification info. However, having to rely on an unofficial verification application is not optimal due to issues with distribution, trust, and usability. In addition, Bluetooth has had multiple security issues in the past.[28] Some of the vulnerabilities are caused by the lengthy specification, which

[28] An estimate of the number of Bluetooth vulnerabilities can be found by searching the list of publicly disclosed security vulnerabilities by the keyword "Bluetooth":`https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Bluetooth`

is difficult to analyse.[29]

We created a second configuration (extended setup) for the voting device to resolve the aforementioned issues. The extended setup consists of ESP32 and a touchscreen. As the touchscreen can be used to display the QR-code and get user input, it was possible to remove the rotary encoder and the unofficial verification application from the extended setup. This way, it is easier to build the device, and the voter can rely on the official vote verification application. The voting device that follows the extended setup is displayed in Figure 22.

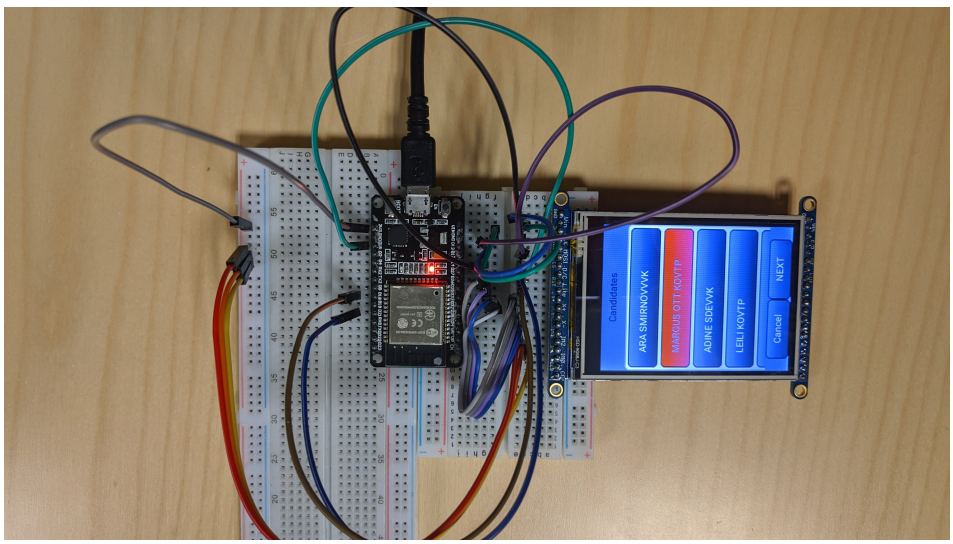


Figure 22. The displayed voting device is built from DOIT ESP32 DevKit v1 and an ILI9341 touchscreen. The image originates from [120].

We used Adafruit's 2.8-inch ILI9341 touchscreen in the extended setup [95, 94]. It has a 240×320 pixel screen resolution, which fits the QR-code. The touchscreen also has a reasonable price tag with ~30 USD for the original Adafruit screen and cheaper options being available for the clones.

Table 8. The pin layout for DOIT DevKit v1 [120].

SPI pins			ILI9341 pins			Touch pins			
CLK	MOSI	CS	D/C	Lite	RST	Y+	Y-	X+	X-
14	13	15	21	5	16	32(A4)	25	26	33(A5)

In the extended setup, the screen is connected to the board over a serial peripheral interface (SPI) so that the board is a master and the screen is a slave. We used the DOIT DevKit v1 board for testing the extended setup. To follow the tested build, the used PIN configuration for the given set-up is given in Table 8. The schematics for the extended setup can be found from in our paper that gives a more detailed overview of how the device was built [120].

[29]The specification for Bluetooth 5.2 contains 3256 pages [150].

5.4.3. Trust base

The main contribution of developing an independent voting client for the Estonian i-voting system comes from increased transparency. Its most visible aspect is the publishing of the source code for the voting client. However, impact-wise, limiting the trust base of the voting client could play a bigger role, as the trusted components could affect vote privacy and vote integrity. An average voter has no means of inspecting the hardware and software running on a general-purpose computer. Thus, voters can not ensure that the trusted components do not contain bugs, vulnerabilities, malware or backdoors.

When analysing the trust base of an official i-voting application, it becomes apparent that the software components of the voting client are a small part of the whole trust base. In addition to the voting software, the voter has to trust the PKI, eID-s, operating system, firmware, third-party software, peripherals connected to the device, and the hardware and software supply chain.

The official voting software is running on a general-purpose computer and is thus affected by its security issues. First of all, it is practically impossible to have a complete overview of the functionalities of the software running on a PC. Trusting the operating system is not sufficient as users install third-party software, which often contains vulnerabilities. Such software can contain bugs in its own code, but also in the libraries, which it depends on. Thus, the supply chain of software also has to be trusted. This can be an issue even when the software vendor is trusted, as the vendor itself is a good target for spreading malware.

Supply chain attacks have become common over the past few years. For example, in 2017, attackers released a malicious update for CCleaner, which was downloaded over two million times until being detected a month later [234]. In the same year, NotPetya malware spread worldwide via a malicious update to the M.E.Doc accounting software [147]. However, the recent incidents with SolarWind [233] and Kaseya [44] have finally started to raise awareness among the wider audience regarding the threats posed by supply chains.

Besides the aforementioned risks, malicious software could also be installed by accident, as an average computer user is not able to check if a downloaded software binary contains a malicious component.

Unfortunately, anti-virus software along with endpoint detection and response systems are not a silver bullet as they can not detect every threat and malware sample [297, 184]. For example, they are not designed to prevent legitimate software from collecting telemetry. Furthermore, anti-virus software on its own is a critical trusted component that runs with elevated privileges. Thus, the anti-virus software vendors must be trusted not to include malicious code, as was suspected in the case of Kaspersky [259, 143].

However, software vendors usually can not guarantee that their software does not contain any bugs and the same holds for anti-virus vendors. There are multiple examples of anti-virus software containing vulnerabilities, many of which have

been discovered by Tavis Ormandy [248, 249, 43, 183]. While vulnerabilities in software that runs with user permissions may not have severe consequences, anti-virus software should be viewed as a parser that runs with administrative permissions, making it a good attack vector.

Some anti-virus vendors add TLS interception functionality to their products by including a self-generated root certificate to the trust store of the local browsers. While such behaviour can simplify the traffic scanning, it also endangers privacy and integrity of the communication protected by TLS. However, monitoring the TLS traffic is not limited to anti-virus products; it is also common in the corporate setting [115]. Thus, it is important to consider the effect that TLS interception and removal can have on i-voting systems [64].

In the case of voting, it is insufficient only to use TLS to protect the integrity and privacy of the exchanged data. Still, TLS is required to prevent the leakage of both identifying information and encrypted votes, which may become decryptable in the future due to the lack of everlasting privacy. Therefore, the official voting client must be configured to prevent traffic from being intercepted. For example, the voting application should pin the certificates that are used to authenticate the server. This has to be done on the application level as it is non-trivial to limit the number of supported root certificates in a general-purpose computer without reducing its usability. Fortunately, such limitations do not hold for a microcontroller-based single-purpose voting device. Thus, to limit the trust base for the microcontroller-based implementation, only one root certificate can be included in its firmware so that it would be possible to initiate a TLS connection to the voting servers. Another option that prevents the TLS traffic from being intercepted is to rely on client certificate authentication [252].

Once the microcontroller is flashed with the voting software, it can be considered a single-purpose device. Thus, it is not affected by unrelated software components, unlike the PC-based voting software. This also limits the number of available communication channels with the outside world for the voting device. Therefore, the risks from the code quality of the unofficial voting client are somewhat reduced as it becomes difficult to deliver an exploit in case vulnerabilities are found in the voting client. Still, the downloading and installing phase of the voting software remains vulnerable to external threats.

That leads to the main part of the trust base of the voting device, namely the software running on the microcontroller. It is not reasonable to independently write all of the required software components for a proof of concept device, especially if open-source alternatives are available. Therefore, we used open-source software components for building the microcontroller-based voting client. The source code for the voting client is available on GitHub [119].

The voting client was built using the Espressif IoT Development Framework (ESP-IDF), created by Espressif Systems. It is an open-source framework built on top of FreeRTOS [301]. ESP-IDF contains libraries and modules that can be selectively enabled in the firmware while developing a new application. The in-

structions for using the ESP-IDF toolchain are available from the ESP-IDF website [299]. In addition, the voting client software also depends on the following libraries:

- `miniz` data compression library [134],
- ESP32 Arduino core [298],
- library for the rotary encoder [26] (basic configuration),
- U8g2 library [192] for the display and UI (basic configuration),
- QRCode library [222] (extended configuration),
- Adafruit Touchscreen library [174] (extended configuration),
- LittlevGL for display connection and UI [189] (extended configuration).

5.4.4. Discussion

Publication of the source code for the unofficial voting client marks the first time that voters can cast an i-vote in Estonia using a system where all components are open-source. While the official voting application remains closed source, the possibility to create an independent voting client shows that voters have an alternative to trusting the official voting application.

The reason for not disclosing the source code of the official voting application was to make the life of the attackers more difficult. Thus, publication of the source code for an unofficial voting client prompts questions about the possible negative side effects. So, could the source code of an unofficial voting client help the attackers in any significant way?

It can be claimed that without the source code of the voting application, the attackers would have difficulties creating malware that interferes with the voting process. However, this claim can be rejected by arguing that the necessary information is already available. The API used to communicate with the vote collection server is described both by the server-side source code and the technical documentation. In addition, a motivated and resourceful attacker has the ability to reverse engineer the official voting application.

Of course, it can be claimed that the server-side source code of the voting system could be published just before the elections start to force the attackers to race against the time. However, by preventing early release of the source code and technical documentation, the positive effects of publishing the source code would disappear, as there would not be sufficient time for independent researchers to conduct an audit.

Furthermore, the preventive measure from the late publication of the source code relies on a few assumptions. First, the attacker must not have gotten access to the source code during the development phase. Second, the voting system must have a different API than the system used during the previous elections. Thus, while the publication of the source code for an unofficial voting client may make it easier to use the vote casting API, it is very likely that a motivated and a resourceful threat actor already has the means to identify the API.

The hardware required for assembling the voting device is relatively cheap, costing between 10-40 US dollars depending on the selected parts. However, it is unclear how much it would cost to build an official voting device that could be distributed to the voters. For example, the security requirements would have to be re-evaluated if such voting devices were distributed or sold to the voters. The design of the proof-of-concept voting device assumes that the voters assemble the device on their own, which significantly lowers the probability that the hardware components would be modified in bulk during delivery or assembly.

However, if voters would not build the devices themselves, they would have to verify that the software running on the device is not malicious. As vote integrity can be verified with the smartphone-based verification application, integrity is not a significant issue in the context of the voting device. However, the voters would somehow have to ensure that the voting device can not leak the vote as this is supposed to be one of the main differences compared to software that runs on a general-purpose computer. It is currently unclear how similar privacy assurances could be achieved on a general-purpose computer without significantly modifying the voting protocol.

While the proof-of-concept voting device could provide an alternative for a limited number of security-conscious voters, it is not ready to be used by the general population.[30] Furthermore, the concept of having to use a personal voting device to protect vote privacy illustrates how difficult it is to build i-voting systems. However, as vulnerable end-user devices remain a critical issue for i-voting systems, it is worth continuing to research whether creating malware-resistant personal voting devices is feasible.

5.5. Should mobile voting be introduced to the Estonian i-voting system?

Along with the diffusion of smartphones, questions have started to appear about the possibility of using mobile devices for voting [322]. It is envisioned that smartphones could replace many of the use cases currently associated with PCs. This has already led to experimental deployments of smartphone-based voting solutions [113, 14, 291].

One argument for introducing a smartphone-based voting client is the need to adapt to the changing environment to increase accessibly and prevent the participation rate from falling. As a counterargument, there are concerns regarding the security of internet voting, which also covers mobile voting [188, 291]. However, the security risks of a specific voting channel have to be evaluated in the context of the whole election system. For example, the risks of introducing a smartphone-based voting client are likely to be more severe if the existing election system

[30]The proof-of-concept voting device was field-tested during the municipal elections in 2021. However, the ballot cast with the device turned out to be invalid and was not included in the tallying phase [209].

does not provide the option to i-vote. If i-voting is already enabled, it has to be analysed how the risk landscape would be affected by introducing a new voting channel. Recent studies show that there are still multiple unsolved issues that complicate the introduction of a smartphone-based voting client [163, 32]. These mainly affect voters' privacy, election integrity, and usability of the voting client.

However, the possibility to cast a vote with a mobile device could also have some advantages. For example, by having more mobility for voting, coercion-resistance could be improved. In addition, the mainstream operating systems used by smartphones are tightly controlled, and mobile devices typically have a smaller attack surface than PC-s [221].

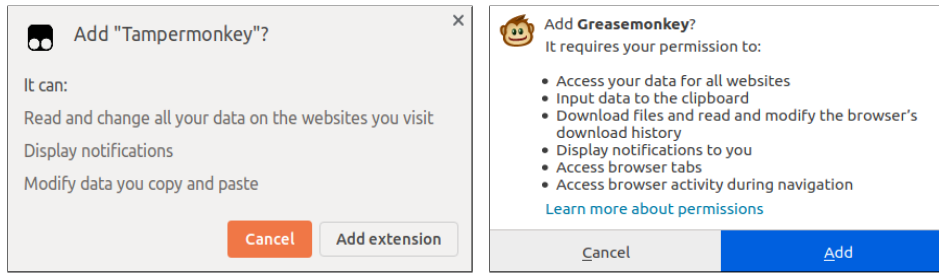
When considering the deployment of a smartphone-based voting solution, there are two ways how a voting client could be delivered to the voter. First, a voting application could be implemented as an interactive web page that could be accessed via a web browser. Second, a voting application could be packaged into a stand-alone Android or iOS application that the voters would have to acquire and install. Both of these approaches have relatively unique implications for the security and privacy of voting.

5.5.1. Browser-based voting client

The simplest way to distribute the voting application is via a dynamic web page. While the possibility to vote from a browser is convenient, it also complicates the security analysis as browsers are not tied to a specific operating system. Thus, by offering the option to participate in elections via a web page, voters can vote from PC-s, smartphones and perhaps even from cars bundled with web browsers [236]. This brings us to the question regarding which browsers should be supported and how to prevent voters from using legacy versions that contain vulnerabilities.

However, even when only the mainstream browsers are supported for voting, there are still significant differences between desktop browsers and smartphone browsers. For example, it is questionable whether ID card could be used in mobile browsers [32]. In addition, two longitudinal studies on security features of mobile browsers revealed that new security features are ported to mobile devices with a long delay, thereby making them less secure than desktop browsers [215, 214]. Furthermore, desktop browsers can be augmented with extensions, which is usually not the case with mobile browsers.

Browser extensions can ask the permission to read and modify the content of the visited web pages, as illustrated by Figure 23. Unlike Firefox, Google Chrome allows users to limit extension permissions for selected websites, but this has to be done manually after the extension is already installed, see Figure 24. However, such an option is not available for extensions installed via enterprise policy, see Figure 25. When considering the impact of browser extensions, it is clear that they could endanger both integrity and secrecy of the vote. Furthermore, this threat is difficult to mitigate as the risk depends on the behaviour of the voter.



(a) Google Chrome extension (desktop) (b) Firefox extension (desktop)

Figure 23. The user has to accept all permissions while installing the extension.

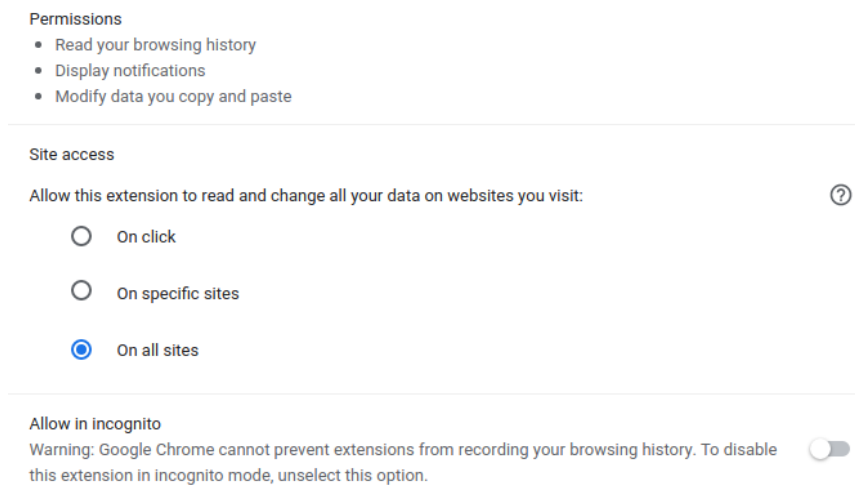


Figure 24. Regular Google Chrome extensions allow users to limit site access manually.

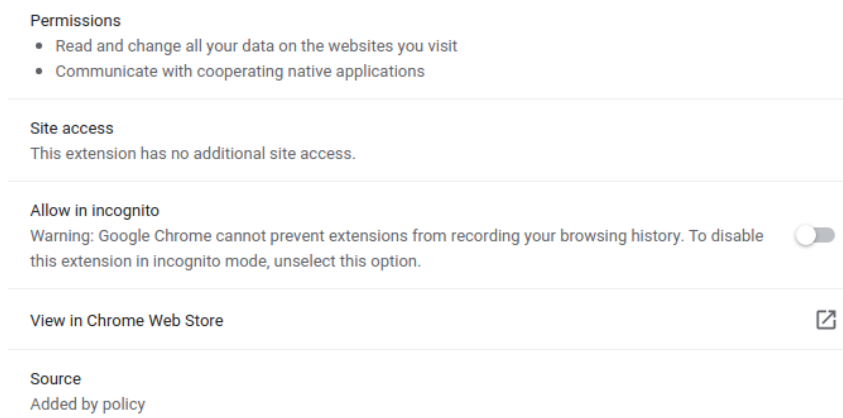


Figure 25. Screenshot of Google Chrome's Token signing extension, which makes it possible to use the Estonian ID card in the browser. The extension is added by policy, which prevents users from limiting site access.

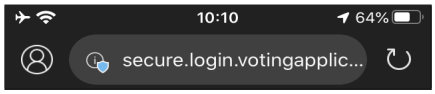
Besides browser extensions, the browsers themselves can leak information about user's behaviour via telemetry. For example, it is common for browsers to analyse the URL-s and downloads to identify phishing sites and malware. In addition, using the URL bar as a search box can leak information. The issues with browser telemetry were studied in 2020 by Leith [208]. The study revealed that Microsoft Edge and Yandex browsers leaked the most information while Brave Browser had the least of such issues. Thus, when considering deploying a voting application via a website, it has to be understood that browsers are not equal in regards to their privacy guarantees.

In addition to the aforementioned risks, also the possibility of network-level traffic interception has to be considered. If the traffic is intercepted, it becomes possible to deliver a modified voting application to the voter, which could influence both the integrity and privacy of the vote. Such an attack could happen in corporate environments where network monitoring is common. By attacking such systems, it becomes possible to target the end-users whose traffic is being intercepted. For example, according to a paper published in 2017, the interception rate of TLS traffic was around 4-11% [115]. A follow-up study conducted by Cloudflare in 2019 revealed that such interception had become more widespread [123]. While previously it was possible to bypass such monitoring by pinning certificates via HTTP Public Key Pinning (HPKP), this is no longer the case as HPKP was removed from browsers. Although Certificate Transparency (CT) can be seen as a replacement that prevents man-in-the-middle attacks, it is only mandatory for certificates issued by publicly-trusted certificate authorities [206]. Thus, CT is not designed nor guaranteed to protect against local interception that uses locally issued certificates [145]. Thereby, if votes are cast using corporate infrastructure, it is difficult to avoid privacy violations, especially in voting systems where TLS is used as the main measure to protect vote privacy [64]. This further highlights that the API-s used in i-voting systems should be designed to be public. Thus, the voting protocol has to use additional cryptographic means to prevent the contents of the queries from being read and modified.

Another significant issue with browser-based voting lies in the widely differing screen sizes. Thus, desktop browsers and mobile browsers have to display candidates differently. Besides the layout of the voting web page, mobile browsers also behave differently than desktop browsers as they attempt to optimise the available screen area. For example, it is common for mobile browsers to hide the URL bar. In addition, as smartphones are commonly used in portrait mode, the URL of the visited web page often does not fit into the URL bar. Thus, web browsers have prioritised showing only part of the URL. However, to prevent phishing attacks, users should be able to view the domain name of the visited web page. Unfortunately, there is historical evidence that mobile browsers have incorrectly displayed long URL-s [215].

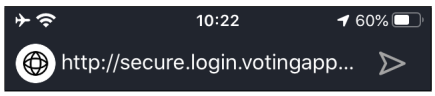
We tested the common mobile browsers to check whether this is still the case. Empirical testing revealed that several browsers did not prioritise displaying the

website's domain in the URL bar. As an outlier, UC Browser on Android displayed the web page's title instead of the URL. Such configuration issues make phishing attacks trivial and further highlight the issues with browser-based voting. The URL bars of the affected browsers are displayed in Figure 26.

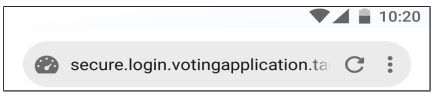


(a) Microsoft Edge (iOS)

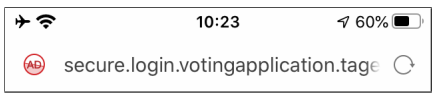
(b) Microsoft Edge (Android)



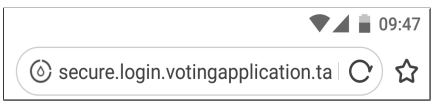
(c) Opera Touch (iOS)



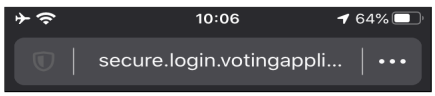
(d) Opera Mini (Android)



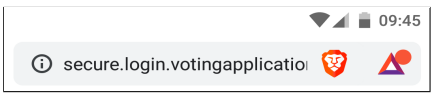
(e) UC Browser (iOS)



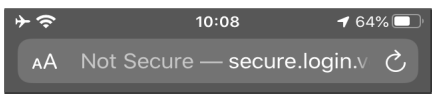
(f) UC Browser (Android)



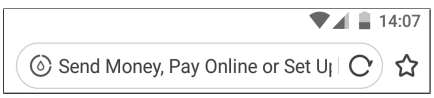
(g) Firefox (iOS)



(h) Brave (Android)



(i) Safari (iOS)



(j) UC Browser (Android) – screenshot is taken while visiting paypal.com

Figure 26. URL bars of the mobile browsers that did not properly display the domain of the website. The screenshots were captured in June 2020. The images a-i display a long URL that contains three subdomains, with the real domain being hidden from the user. Surprisingly, UC Browser on Android displayed the title of the page instead of the URL.

5.5.2. Standalone voting application

Another way to implement the voting client is to create a native Android or iOS application. These applications are by default distributed via official application stores in the form of signed package files.

Compared to the voting application distributed as a web page, it would be possible to support the ID card for authenticating voters and signing ballots [32]. In addition, the native application provides more options to control security of the voting environment. For example, both iOS and Android applications can pin certificates to prevent man-in-the-middle attacks. In addition, official application stores provide an option to limit the devices where the application can be installed. Thereby, it is possible to prevent insecure legacy devices from being able to install the voting application.

While the aforementioned restrictions would increase the security level, they would also prevent some eligible voters from using the mobile voting channel. In

addition, by limiting the distribution of the official voting application, an incentive could be created for building an independent voting client. However, unofficial voting applications may create new security issues as they are unlikely to get a similar scrutiny level as the official voting application. In addition, the ability to choose among multiple voting applications increases the risk of a fake or malicious voting application being distributed.

While deploying the voting application via an official app store is convenient, it also has downsides. For example, the app store owner has the power to decide whether to accept or remove the application from the app store. It is also possible to remotely remove applications from end-user devices [221]. In addition, the owner of the app store gets access to the profiles of voters who have installed the voting or verification applications. If such information leaks into the hands of malicious parties, it could be used to launch targeted malware attacks with the aim of changing the election outcome. For example, by only infecting the desktop computers of users who have not installed a verification application, it is less likely for the attack to be detected.

Furthermore, research has shown that Google Play Store has issues with applications being signed with weak keys [323] and applications being signed with legacy APK signature schemes [163]. Interestingly, Google requires the certificates for keys that were used to sign the applications to be valid at least until October 22nd 2033 [118]. As the developer signing key is needed to release updates, rotating the keys without creating a new application is difficult. Thus, key rotation functionality was added with the introduction of APK Signature scheme v3, released in August 2018 [144]. However, it is not mandatory to use this APK signature scheme as it is not compatible with versions older than Android 9.

While previously the signing key of an Android application was under the control and responsibility of the developer, this is going to change. In 2017, Google introduced a new application signing functionality called Google Play App Signing, allowing Google to store and handle the signing key [137]. In 2020, Google announced that App Bundles would replace the APK format in 2021 [116]. However, to use App Bundle, Google Play App Signing must be used, which effectively ends developers control over the signing keys [117, 23]. As a compensation mechanism, Google offers developers the option to use another signing key for Code Transparency, allowing developers to check that Google has not tampered with the source code. However, the check would have to be performed manually as Android does not check the Code Transparency signature at install time [146]. In addition, the documentation for Code Transparency mentions that only DEX files and native libraries are covered under the signature, which leaves other files without a guarantee.

Although it could be claimed that to release an update to the application, both the signing key and upload key are required, it is actually not the case as Google can reset the upload key [137]. Therefore, when both of these keys are accessible to Google, it is in principle possible to release updates without the permission of

the application developer. However, this is something that should not be allowed for a critical application used in elections. The election organiser must control the signing key for the voting application, as otherwise voting application's integrity could be questioned.

5.5.3. Discussion

Both the PC-based voting applications and their smartphone-based counterparts are vulnerable to malware. However, getting administrative access to the device is slightly more difficult on mobile devices than common PC-based operating systems. Regardless of whether a web browser or a native application is used to cast a vote, if the device is infected, malware on the device could violate vote privacy. The only way to counter such a threat would be to rely on code voting, which would require a complete overhaul of the voting protocol.

When considering the risk profile of a web application-based voting client, it has to be understood that browsers themselves significantly increase the attack surface compared to native applications.

When leaving aside the need to trust the end-user device, the properties of different types of voting applications can be compared. For example, it is easier for voters to verify the integrity of a PC-based voting application as it is possible to verify digital signatures and compare hash values of the corresponding binary. The election organiser signs the Estonian i-voting application, and both Windows and macOS automatically verify the signature.[31] However, when the voting application would be delivered as a web application, it would be difficult for voters to ensure that valid source code was sent to their browsers.

Although Subresource Integrity (SRI) could be used to check the integrity of the code fetched from the server, it does not protect against compromised servers and man-in-the-middle attacks [18]. Furthermore, when considering mobile browsers, is it rare to have an option to view information about the TLS configuration and certificate [163]. When combining the lack of information regarding the certificate and the issues with displaying long URL-s in mobile browsers, it becomes apparent that phishing could become a real problem when a web application-based voting client is introduced.

Similarly, it is difficult for regular voters to verify the integrity of native Android and iOS applications. Therefore, they would have to trust that a valid and non-tampered application is delivered from the application store. However, the necessity to trust parts of the voting system or its supporting systems goes against the aim of making the voting system transparent and verifiable. Thus, additional trust assumptions should not be introduced into the voting system as these can be seen as possible attack vectors.

Many of the aforementioned issues are related to the limited screen area avail-

[31] While the operating system can automatically verify that a binary is signed, a human must check that the correct entity signed it.

able on smartphones, which has resulted in optimisations that have a negative impact on security. However, in addition to the limited screen area, it would also be difficult for voters to verify the authenticity of smartphone-based voting applications regardless of the deployment methods. For example, counterfeit applications have been found both from Play Store and iOS App Store [139, 268]. Thus, there is a risk that voters could be tricked into using fake voting applications.

Smartphone-based voting also introduces the question of whether the existing method for checking individual verifiability has to be modified. By using a PC-based voting application, the voter can verify the vote on a smartphone. However, if the vote is cast on a smartphone, a voter may not have access to a second smartphone to verify the vote. Thus, introducing smartphone-based voting has an impact on the existing vote verification system [32].

Due to many security issues, it is currently not recommended to introduce mobile voting to the Estonian i-voting system. However, it has to be taken into account that by not providing an official smartphone-based voting application, an incentive is created to develop an unofficial voting client for mobile devices. Thus, if the technical shortcomings get fixed, the situation has to be re-evaluated. However, even when the technical issues are solved, it remains unclear how mobile voting would affect individual verifiability and coercion-resistance. Further studies have to be conducted to answer these questions.

6. CONCLUSION

This thesis focused on the privacy and coercion-resistance properties of voting systems. We described new attacks against traditional paper-based voting systems, analysed the applicability of existing anti-coercion measures and proposed mitigation measures for some of the issues present in the Estonian i-voting system.

The security guarantees offered by election systems are not constant as they are affected by changes in society and by the advancement of technology. For example, the privacy assurances provided by polling booths were significantly lowered by the adoption of smartphones. However, advances in cryptography have made it possible to increase integrity guarantees by using end-to-end verifiable voting systems.

As the first contribution of this thesis, we highlighted the impact of technology by creating two new proof-of-concept attacks against vote privacy in the traditional paper-based voting system. The attacks utilise an acoustic side-channel caused by the acoustic emanations of the voter filling in the ballot. The attacks that bypass the protective environment provided by the polling booth prompt the question of where to set the baseline when considering the level of coercion-resistance that is achievable in practice. Determining a sufficient level of coercion-resistance becomes more and more relevant with the push towards end-to-end verifiable voting systems.

As the second contribution of this thesis, we studied a selection of modern i-voting systems to find the optimal method for providing coercion-resistance in the remote setting. The research revealed that the systems that attempt to simultaneously provide integrity and privacy assurances rely on assumptions that are difficult to apply in practice. While re-voting is a rare example of an easily usable anti-coercion measure, it also has its downsides when considering the verifiability properties of the election system. Thus, finding the balance between the conflicting security requirements ultimately leads to a political or legal decision.

Security requirements set for a voting system have to be viewed in the context of the surrounding environment. This also holds for the Estonian i-voting system, which targets one possible equilibrium between integrity and coercion-resistance properties. As the third contribution, we described issues that affected the voting and verification protocol used by the Estonian i-voting system. In addition, we proposed mitigation measures that could be applied within the existing i-voting protocol.

One of the central issues with remote voting is the untrustworthiness of voters' computers. Malware on the voter's computer has the potential to jeopardise both vote privacy and vote integrity. Thus, as the fourth contribution, we described how a microcontroller-based voting client could be built for the Estonian i-voting system. The proof-of-concept device only runs the voting client, which significantly reduces its attack surface. Along with the device's description and security analysis, we also published the source code for the voting client. This marked the

first time that the source code for the Estonian i-voting client was made available.

As the fifth and final contribution, we assessed the potential security impact of creating a smartphone-based voting client. In general, there are two possibilities for deploying such a voting option. First, the voting client can be presented to the voter in the form of a dynamic web page. In this case, the voter would have to rely on a web browser to navigate to the voting website. Alternatively, the voting client could be packaged into an independent application delivered to the voters via official application stores. Both of these options have their upsides and downsides. However, the website-based voting client clearly stood out due to the number of additional security risks. Many of the risks are inherent to smartphone-based browsers, which means that the security landscape must be reassessed once the issues have been resolved.

BIBLIOGRAPHY

[1] Civil Rights Act of 1960, § 301, 52 U.S.C. § 20701. Available at: https://www.govinfo.gov/app/details/USCODE-2019-title52/USCODE-2019-title52-subtitleII-chap207-sec20701 (Accessed: 10.05.2021).

[2] *Hansard's Parliamentary Debates*, volume 187 of *3rd series*, pages 1305–1306. 30 May 1867. Digitised edition available at: https://api.parliament.uk/historic-hansard/commons/1867/may/30/committee-progress-may-28#column_1305.

[3] Identity Documents Act. *Riigi Teataja*, RT I(25), February 1999. English translation available at https://www.riigiteataja.ee/en/eli/ee/528122020004. Accessed: 2021-03-22.

[4] Digital Signatures Act. *Riigi Teataja*, RT I(26), December 2000. English translation available at https://www.riigiteataja.ee/en/eli/530102013080. Accessed: 2021-07-01.

[5] Penal Code. *Riigi Teataja*, RT I(61), September 2002. English translation available at https://www.riigiteataja.ee/en/eli/ee/522012015002. Accessed: 2021-09-21.

[6] Riigikogu Election Act. *Riigi Teataja*, RT I(57), June 2002. §77[1] covers preservation of ballot papers and election documents. English translation available at https://www.riigiteataja.ee/en/eli/ee/514122020002#para77b1. Accessed: 2021-09-21.

[7] Riigikogu Election Act. *Riigi Teataja*, RT I(57), June 2002. English translation available at https://www.riigiteataja.ee/en/eli/ee/514122020002. Accessed: 2021-09-21.

[8] Local Government Council Election Act (in Estonian). *Riigi Teataja*, RT I(36), September 2005. Amendment to the Local Government Council Election Act. https://www.riigiteataja.ee/akt/938781. Accessed: 2021-05-10.

[9] Riigikogu Election Act (in Estonian). *Riigi Teataja*, RT I(57), July 2006. Amendment to the Riigikogu Election Act. https://www.riigiteataja.ee/akt/1045561. Accessed: 2021-05-10.

[10] A law to change the European Parliament Election Act and other laws (in Estonian). *Riigi Teataja*, RT I(1), July 2018. https://www.riigiteataja.ee/akt/109072018001. Accessed: 2021-03-22.

[11] Michel Abdalla, Tor Erling Bjørstad, Carlos Cid, Benedikt Gierlichs, Andreas Hülsing, Atul Luykx, Kenneth G. Paterson, Bart Preneel, Ahmad-Reza Sadeghi, Terence Spies, Martijn Stam, Michael Ward, Bogdan Warinschi, and Gaven Watson. Algorithms, Key Size and Protocols Report. Technical report, ECRYPT CSA, 2018.

[12] Michel Abdalla, Tor Erling Bjørstad, Carlos Cid, Benedikt Gierlichs, Andreas Hülsing, Atul Luykx, Kenneth G. Paterson, Bart Preneel, Ahmad-Reza Sadeghi, Terence Spies, Martijn Stam, Michael Ward, Bogdan Warinschi, and Gaven Watson. Algorithms, Key Size and Protocols Report. Technical report, ECRYPT CSA, 2018.

[13] Lillian Ablon and Andy Bogart. *Zero Days, Thousands of Nights: The Life and Times of Zero-Day Vulnerabilities and Their Exploits*. RAND Corporation, 2017.

[14] Abigail Abrams. Smartphone Voting Could Expand Accessibility, But Election Experts Raise Security Concerns. 2019. `https://time.com/5717479/mobile-voting-accessibility/`. Accessed: 2021-07-15.

[15] Ben Adida. Helios: Web-based Open-Audit Voting. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 335–348. USENIX Association, 2008.

[16] Arduino AG. Analogread. `https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/`. Accessed: 2021-12-05.

[17] National Security Agency. Commercial National Security Algorithm (CNSA) Suite, August 2015. `https://apps.nsa.gov/iaarchive/programs/iad-initiatives/cnsa-suite.cfm`. Accessed: 2021-12-04.

[18] Devdatta Akhawe, Frederik Braun, François Marier, and Joel Weinberger. Subresource Integrity, 2016. `https://www.w3.org/TR/SRI/`. Accessed: 2021-07-16.

[19] Monjur Alam, Haider Adnan Khan, Moumita Dey, Nishith Sinha, Robert Locke Callan, Alenka G. Zajic, and Milos Prvulovic. One&Done: A Single-Decryption EM-Based Attack on OpenSSL's Constant-Time Blinded RSA. In William Enck and Adrienne Porter Felt, editors, *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 585–602. USENIX Association, 2018.

[20] Bethany Albertson and Kimberly Guiler. Conspiracy Theories, Election Rigging, and Support for Democratic Norms. *Research & Politics*, 7(3):447–460, 2020.

[21] Syed Taha Ali and Judy Murray. An Overview of End-to-End Verifiable Voting Systems. In Feng Hao and Peter Y. A. Ryan, editors, *Real-World Electronic Voting: Design, Analysis and Deployment*, pages 173–217. CRC Press, 2017.

[22] Joël Alwen, Rafail Ostrovsky, Hong-Sheng Zhou, and Vassilis Zikas. Incoercible Multi-party Computation and Universally Composable Receipt-Free Voting. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 763–780, 2015.

[23] Ron Amadeo. Google Play dumps APKs for the more Google-controlled "Android App Bundle", 2021. https://arstechnica.com/gadgets/2021/07/google-play-dumps-apks-for-the-more-google-controlled-android-app-bundle/. Accessed: 2021-07-16.

[24] Arne Ansper, Ahto Buldas, Mart Oruaas, Jaan Priisalu, Anto Veldre, Jan Willemson, and Kaur Virunurm. E-voting concept security: analysis and measures, 2003. Estonian National Electoral Commitee, EH-02-01. https://web.archive.org/web/20040928014413/http://www.vvk.ee/elektr/docs/Analyys-01.pdf. Accessed: 2021-07-01.

[25] Arne Ansper, Ahto Buldas, and Jan Willemson. Cryptographic algorithms lifecycle report 2017, 2018. Cybernetica research report A-101-9, https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/cryptographic-algorithms-lifecycle-report-2017.pdf.

[26] David Antliff. esp32-rotary-encoder. https://github.com/DavidAntliff/esp32-rotary-encoder. Accessed: 2021-11-30.

[27] Andrew W Appel, Maia Ginsburg, Harri Hursti, Brian W Kernighan, Christopher D Richards, and Gang Tan. Insecurities and inaccuracies of the Sequoia AVC Advantage 9.00 H DRE voting machine, 2008.

[28] Andrew W. Appel, Maia Ginsburg, Harri Hursti, Brian W. Kernighan, Christopher D. Richards, Gang Tan, and Penny Venetis. The New Jersey Voting-machine Lawsuit and the AVC Advantage DRE Voting Machine. In *2009 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '09*. USENIX Association, 2009.

[29] Diego F. Aranha and Jeroen van de Graaf. The Good, the Bad, and the Ugly: Two Decades of E-Voting in Brazil. *IEEE Secur. Priv.*, 16(6):22–30, 2018.

[30] Roberto Araújo, Narjes Ben Rajeb, Riadh Robbana, Jacques Traoré, and Souheib Yousfi. Towards Practical and Secure Coercion-Resistant Electronic Elections. In *CANS 2010, Proceedings*, volume 6467 of *LNCS*, pages 278–297. Springer, 2010.

[31] Luca Arduser, Pascal Bissig, Philipp Brandes, and Roger Wattenhofer. Recognizing text using motion data from a smartwatch. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2016, Sydney, Australia, March 14-18, 2016*, pages 1–6. IEEE Computer Society, 2016.

[32] Cybernetica AS. Mobile voting feasibility study and risk analysis, 2020. Cybernetica research report T-184-5, https://www.valimised.ee/sites/default/files/uploads/eng/2020_m-voting-report.pdf. Accessed: 2021-03-22.

[33] Cybernetica AS. Applying biometric facial recognition for electronic voting (in Estonian), 2021. Cybernetica research report D-16-12, https:

//www.ria.ee/sites/default/files/content-editors/publikatsioonid/biomeetria-aruanne.pdf. Accessed: 2021-08-17.

[34] GN Audio A/S. Jabra Speak 410 Datasheet, 2018. https://www.jabra.com/_/media/Jabra_VXi_Product-Documentation/Jabra-SPEAK-410-Series/Datasheets/RevA/Jabra-Speak-410-Datasheet-A4-screen.pdf. Accessed: 2021-10-26.

[35] Information System Authority. Estonia ID1 Chip/App 2018, Technical Description, 2020. Document Release Version: V0.9, https://installer.id.ee/media/id2019/TD-ID1-Chip-App.pdf.

[36] Information System Authority. Means of eID. https://www.ria.ee/en/state-information-system/electronic-identity-eid/means-eid.html, 2021. Accessed: 2021-07-01.

[37] Adam J. Aviv, Pavol Cerný, Sandy Clark, Eric Cronin, Gaurav Shah, Micah Sherr, and Matt Blaze. Security Evaluation of ES&S Voting Machines and Election Management System. In *2008 USENIX/ACCURATE Electronic Voting Workshop, EVT 2008*. USENIX Association, 2008.

[38] Jan Baar, Jan Tippner, and Vladimír Gryc. The influence of wood density on longitudinal wave velocity determined by the ultrasound method in comparison to the resonance longitudinal method. *European Journal of Wood and Wood Products*, 70(5):767–769, September 2012.

[39] Michael Backes, Markus Dürmuth, Sebastian Gerling, Manfred Pinkal, and Caroline Sporleder. Acoustic Side-Channel Attacks on Printers. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 307–322. USENIX Association, 2010.

[40] J. Bannet, D. W. Price, A. Rudys, J. Singer, and D. S. Wallach. Hack-a-vote: Security issues with electronic voting systems. *IEEE Security & Privacy*, 2(1):32–37, 2004.

[41] Pablo Barberá, John T. Jost, Jonathan Nagler, Joshua A. Tucker, and Richard Bonneau. Tweeting From Left to Right: Is Online Political Communication More Than an Echo Chamber? *Psychological Science*, 26(10):1531–1542, 2015.

[42] Elaine Barker. NIST Special Publication 800-57 Recommendation for Key Management – Part 1: General (Revision 5), 2020. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf. Accessed: 2021-03-22.

[43] Brian Barrett. A Vicious Microsoft Bug Left a Billion PCs Exposed. 2017. https://www.wired.com/2017/05/vicious-microsoft-bug-left-billion-pcs-exposed/. Accessed: 2021-07-14.

[44] Brian Barrett. A New Kind of Ransomware Tsunami Hits Hundreds of Companies. *Wired*, June 2021. https://www.wired.com/story/kaseya-supply-chain-ransomware-attack-msps/.

[45] Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical Multi-Candidate Election System. In Ajay D. Kshemkalyani and Nir Shavit, editors, *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, Newport, Rhode Island, USA, August 26-29, 2001*, pages 274–283. ACM, 2001.

[46] Pavlo Bekhta, Peter Nimez, and Ladislav Kucera. The study of sound propagation in the wood-based composite materials. In *International Symposium on Nondestructive Testing of Wood*, volume 12, pages 33–41, 2000.

[47] Josh Benaloh. Simple Verifiable Elections. In *2006 USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'06, Vancouver, BC, Canada, August 1, 2006*, 2006.

[48] Josh Benaloh. Rethinking Voter Coercion: The Realities Imposed by Technology. In *2013 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '13, Washington, D.C., USA, August 12-13, 2013*. USENIX Association, 2013.

[49] Josh Benaloh. STROBE-Voting: Send Two, Receive One Ballot Encoding. In Robert Krimmer, Melanie Volkamer, David Duenas-Cid, Oksana Kulyk, Peter B. Rønne, Mihkel Solvak, and Micha Germann, editors, *Electronic Voting - 6th International Joint Conference, E-Vote-ID 2021, Virtual Event, October 5-8, 2021, Proceedings*, volume 12900 of *Lecture Notes in Computer Science*, pages 33–46. Springer, 2021.

[50] Josh Benaloh, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, and Poorvi L. Vora. End-to-end verifiability. *arXiv preprint arXiv:1504.03778*, 2015. `https://arxiv.org/abs/1504.03778`.

[51] Josh Cohen Benaloh and Dwight Tuinstra. Receipt-Free Secret-Ballot Elections (Extended Abstract). In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 544–553, 1994.

[52] Yigael Berger, Avishai Wool, and Arie Yeredor. Dictionary Attacks Using Keyboard Acoustic Emanations. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, pages 245–254. ACM, 2006.

[53] Matthew Bernhard. *Election Security is Harder Than You Think*. Ph.D. Dissertation, University of Michigan, 2020. `https://deepblue.lib.umich.edu/bitstream/handle/2027.42/163272/matber_1.pdf`. Accessed: 2021-03-22.

[54] Matthew Bernhard, Josh Benaloh, J. Alex Halderman, Ronald L. Rivest, Peter Y. A. Ryan, Philip B. Stark, Vanessa Teague, Poorvi L. Vora, and Dan S. Wallach. Public Evidence from Secret Ballots. In Robert Krimmer,

Melanie Volkamer, Nadja Braun Binder, Norbert Kersting, Olivier Pereira, and Carsten Schürmann, editors, *Electronic Voting - Second International Joint Conference, E-Vote-ID 2017, Bregenz, Austria, October 24-27, 2017, Proceedings*, volume 10615 of *Lecture Notes in Computer Science*, pages 84–109. Springer, 2017.

[55] Abram C. Bernheim. The Ballot in New York. *Political Science Quarterly*, 4(1):130–152, 1889.

[56] Rachel Bittner, Eric Humphrey, and Juan Bello. PySOX: Leveraging the Audio Signal Processing Power of SOX in Python. International Conference on Music Information Retrieval (ISMIR-16), 2016.

[57] Peter Brent. The Australian ballot: Not the secret ballot. *Australian Journal of Political Science*, 41(1):39–50, 2006.

[58] Estonian Public Broadcasting. E-Voting Source Code Made Public. *ERR News*, June 2013. `https://news.err.ee/107779/e-voting-source-code-made-public`. Accessed: 2021-11-30.

[59] Ahto Buldas, Kristo Heero, Peeter Laud, Riivo Talviste, and Jan Willemson. Cryptographic algorithms lifecycle report 2016. Cybernetica research report A-101-3, `https://www.ria.ee/sites/default/files/content-editors/publikatsioonid/cryptographic_algorithms_lifecycle_report_2016.pdf`.

[60] Joseph A. Calandrino and William Clarkson. Some Consequences of Paper Fingerprinting for Elections. In David Jefferson, Joseph Lorenzo Hall, and Tal Moran, editors, *2009 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '09, Montreal, Canada, August 10-11, 2009*. USENIX Association, 2009.

[61] Jan Camenisch and Victor Shoup. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, pages 126–144, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[62] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable Encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 1997.

[63] Rafael Capurro. Privacy. An Intercultural Perspective. *Ethics and Information Technology*, 7(1):37–47, 2005.

[64] Anthony Cardillo and Aleksander Essex. The Threat of SSL/TLS Stripping to Online Voting. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - Third International Joint Conference, E-Vote-ID 2018, Bregenz, Austria, October 2-5, 2018, Proceedings*, volume 11143 of *Lecture Notes in Computer Science*, pages 35–50. Springer, 2018.

[65] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. BeleniosRF: A Non-interactive Receipt-Free Electronic Voting Scheme. In *Proceedings of 2016 ACM CCS*, pages 1614–1625, New York, NY, USA, 2016. ACM.

[66] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM*, 24(2):84–88, 1981.

[67] David Chaum. Surevote: Technical Overview. In *Proceedings of the workshop on trustworthy elections (WOTE'01)*, 2001.

[68] David Chaum, Aleksander Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan T. Sherman, and Poorvi L. Vora. Scantegrity: End-to-End Voter-Verifiable Optical-Scan Voting. *IEEE Secur. Priv.*, 6(3):40–46, 2008.

[69] Emily Chen, Ashok Deb, and Emilio Ferrara. #Election2020: the first public Twitter dataset on the 2020 US Presidential election. *Journal of Computational Social Science*, 2021.

[70] Benoît Chevallier-Mames, Pierre-Alain Fouque, David Pointcheval, Julien Stern, and Jacques Traoré. On Some Incompatible Properties of Voting Schemes. In *Towards Trustworthy Elections, New Directions in Electronic Voting*, pages 191–199, 2010.

[71] S. Choi, A. S. Lee, and S. Lee. On-Line Handwritten Character Recognition with 3D Accelerometer. In *2006 IEEE International Conference on Information Acquisition*, pages 845–850, 2006.

[72] Catalin Cimpanu. Hacker steals government ID database for Argentina's entire population. *The Record*, October 2021. `https://therecord.media/hacker-steals-government-id-database-for-argentinas-entire-population/`. Accessed: 2021-10-27.

[73] Jeremy Clark and Urs Hengartner. Panic Passwords: Authenticating under Duress. In *HotSec'08, Proceedings*. USENIX Association, 2008.

[74] Jeremy Clark and Urs Hengartner. Selections: Internet Voting with Over-the-Shoulder Coercion-Resistance. In George Danezis, editor, *FC 2011, Revised Selected Papers*, volume 7035 of *LNCS*, pages 47–61. Springer, 2011.

[75] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a Secure Voting System. In *2008 IEEE Symposium on Security and Privacy (S&P 2008), 18-21 May 2008, Oakland, California, USA*, pages 354–368. IEEE Computer Society, 2008.

[76] Josh D. Cohen and Michael J. Fischer. A Robust and Verifiable Cryptographically Secure Election Scheme (Extended Abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 372–382. IEEE Computer Society, 1985.

[77] Tim Colebatch. Metre-long ballot paper means voters will need to read the fine print. *The Sydney Morning Herald*, August 2013. https://www.smh.com.au/politics/federal/metre-long-ballot-paper-means-voters-will-need-to-read-the-fine-print-20130817-2s3yw.html. Accessed: 2021-10-27.

[78] Estonian National Electoral Committee. E-voting in Linux operating system (in Estonian), 2005. https://web.archive.org/web/20051210183208/http://www.valimised.ee/linux.html. Accessed: 2021-10-20.

[79] Estonian National Electoral Committee. E-voting in MacOS X operating system (in Estonian), 2005. https://web.archive.org/web/20051210183916/http://www.valimised.ee/macos.html. Accessed: 2021-10-20.

[80] Estonian National Electoral Committee. E-voting in Windows operating system (in Estonian), 2005. https://web.archive.org/web/20051210192254/http://www.valimised.ee/windows.html. Accessed: 2021-10-20.

[81] Estonian National Electoral Committee. E-voting in MacOS X operating system (in Estonian), 2007. https://web.archive.org/web/20070206195425/http://www.valimised.ee/macos.html. Accessed: 2021-10-20.

[82] Estonian National Electoral Committee. Voting in Windows operating system (in Estonian), 2009. https://web.archive.org/web/20091001074000/http://www.valimised.ee:80/windows.html. Accessed: 2021-10-20.

[83] Estonian National Electoral Committee. E-Voting System. General Overview. Estonian version is available from: http://www.vvk.ee/public/dok/Uldkirjeldus_e_haaletamine_092010.pdf, 2010. Accessed: 2021-07-01.

[84] Estonian National Electoral Committee. E-Voting System. General Overview. https://www.valimised.ee/sites/default/files/uploads/eng/General_Description_E-Voting_2010.pdf, 2010. Accessed: 2021-11-30.

[85] Estonian National Electoral Committee. How to vote online (in Estonian), 2011. https://web.archive.org/web/20110720131557/http://www.valimised.ee/internet.html. Accessed: 2021-10-20.

[86] Estonian National Electoral Committee and the State Electoral Office. Checking of an i-vote. https://www.valimised.ee/en/internet-voting/guidelines/checking-i-vote. Accessed: 2021-09-29.

[87] Estonian National Electoral Committee and the State Electoral Office. Requirements to the voter and their computer. https://www.valimised.

ee/en/internet-voting/guidelines/requirements-voter-and-their-computer. Accessed: 2021-07-08.

[88] Estonian National Electoral Committee and the State Electoral Office. Statistics about Internet voting in Estonia. https://www.valimised.ee/en/archive/statistics-about-internet-voting-estonia. Accessed: 2021-07-06.

[89] Estonian National Electoral Committee and the State Electoral Office. Instruction for the Preparation of Configurations in IVXV (in Estonian), 2019. Document version IVXV-JSK-1.5.0, https://www.valimised.ee/sites/default/files/uploads/eh/IVXV-seadistuste-koostejuhend.pdf. Accessed: 2021-09-21.

[90] Estonian National Electoral Committee and the State Electoral Office. IVXV architecture. Technical report, 2019. Document IVXV-AR-EN-1.4.0, https://www.valimised.ee/sites/default/files/2021-05/IVXV%20architecture%20%E2%80%93%20overview%20of%20technical%20realisation.pdf. Accessed: 2021-07-02.

[91] Estonian National Electoral Committee and the State Electoral Office. IVXV key application (in estonian). Technical report, 2019. Document IVXV-SVR-1.4.0, https://www.valimised.ee/sites/default/files/uploads/eh/IVXV-votmerakendus.pdf. Accessed: 2021-03-22.

[92] Estonian National Electoral Committee and the State Electoral Office. Stages of i-voting in voter application, 2021. http://web.archive.org/web/20211007032800/https://www.valimised.ee/en/internet-voting/guidelines/stages-i-voting-voter-application. Accessed: 2021-10-20.

[93] Estonian National Electoral Committee and the State Electoral Office (in Estonian). Statistics about Internet voting in Estonia. https://www.valimised.ee/valimiste-arhiiv/elektroonilise-haaletamise-statistika. Accessed: 2021-11-30.

[94] ON TAT INDUSTRIAL COMPANY. Specification for Product Model DT280QV10-CT (Rev.B), 2014. https://cdn-learn.adafruit.com/assets/assets/000/035/819/original/SPEC-DT280QV10-CT_Rev.B.pdf. Accessed: 2021-07-13.

[95] ILI TECHNOLOGY CORP. ILI9341 Specification V1.11. https://cdn-shop.adafruit.com/datasheets/ILI9341.pdf. Accessed: 2021-07-13.

[96] Swiss Broadcasting Corporation. Switzerland's open-air voting is now streamed online, 2019. https://www.swissinfo.ch/eng/directdemocracy/landsgemeinde-live_switzerland-s-open-air-voting-is-now-streamed-online-/44922306. Accessed: 2021-10-04.

[97] Véronique Cortier, Fabienne Eigner, Steve Kremer, Matteo Maffei, and Cyrille Wiedling. Type-Based Verification of Electronic Voting Protocols.

In *Principles of Security and Trust - 4th International Conference, POST 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015, Proceedings*, pages 303–323, 2015.

[98] Véronique Cortier, David Galindo, Ralf Küsters, Johannes Müller, and Tomasz Truderung. SoK: Verifiability Notions for E-Voting Protocols. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 779–798. IEEE Computer Society, 2016.

[99] Thomas M. Cover and Peter E. Hart. Nearest Neighbor Pattern Classification. *IEEE Trans. Inf. Theory*, 13(1):21–27, 1967.

[100] Ronald Cramer, Matthew K. Franklin, Berry Schoenmakers, and Moti Yung. Multi-Autority Secret-Ballot Elections with Linear Work. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 72–83. Springer, 1996.

[101] Braden L. Crimmins, Marshall Rhea, and J. Alex Halderman. RemoteVote and SAFE Vote: Towards Usable End-to-End Verification for Vote-by-Mail. *arXiv preprint arXiv:2111.08662*, 2021. `https://arxiv.org/abs/2111.08662`.

[102] D. Crocker, T. Hansen, and M. Kucherawy. DomainKeys Identified Mail (DKIM) Signatures. STD 76, RFC Editor, September 2011. `http://www.rfc-editor.org/rfc/rfc6376.txt`.

[103] MALCOLM CROOK and TOM CROOK. The Advent of the Secret Ballot in Britain and France, 1789–1914: From Public Assembly to Private Compartment. *History*, 92(308):449–471, 2007.

[104] Chris Culnane, Aleksander Essex, Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. Knights and Knaves Run Elections: Internet Voting and Undetectable Electoral Fraud. *IEEE Security & Privacy*, 17(4):62–70, 2019.

[105] Olivier de Marneffe, Olivier Pereira, and Jean-Jacques Quisquater. Electing a University President Using Open-Audit Voting: Analysis of Real-World Use of Helios. In David Jefferson, Joseph Lorenzo Hall, and Tal Moran, editors, *2009 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '09, Montreal, Canada, August 10-11, 2009*. USENIX Association, 2009.

[106] Gerson de Souza Faria and Hae Yong Kim. Differential Audio Analysis: A New Side-Channel Attack on PIN Pads. *International Journal of Information Security*, 18(1):73–84, 2019.

[107] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Coercion-Resistance and Receipt-Freeness in Electronic Voting. In *19th IEEE Computer Secu-*

rity Foundations Workshop, (CSFW-19 2006), 5-7 July 2006, Venice, Italy, pages 28–42, 2006.

[108] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying Privacy-type Properties of Electronic Voting Protocols. *J. Comput. Secur.*, 17(4):435–487, 2009.

[109] Bella M DePaulo and Deborah A Kashy. Everyday Lies in Close and Casual Relationships. *Journal of Personality and Social Psychology*, 74(1):63–79, 1998.

[110] Schweizerische Depeschenagentur. Thousands of ballot papers land on the streets in Geneva, 2021. `https://www.tagesanzeiger.ch/tausende-stimmzettel-landen-in-genf-auf-der-strasse-181808024476`. Accessed: 2021-10-14.

[111] David L. Dill, Bruce Schneier, and Barbara Simons. Voting and Technology: Who Gets to Count Your Vote? *Commun. ACM*, 46(8):29–31, 2003.

[112] National Security Agency Research Directorate. Ghidra Software Reverse Engineering Framework. `https://github.com/NationalSecurityAgency/ghidra`. Accessed: 2021-07-13.

[113] Emily Dreyfuss. Smartphone Voting Is Happening, but No One Knows if It's Safe. *Wired*, August 2018. `https://www.wired.com/story/smartphone-voting-is-happening-west-virginia/`.

[114] Jonathan Driedger and Meinard Müller. A Review of Time-Scale Modification of Music Signals. *Applied Sciences*, 6(2), 2016.

[115] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J. Alex Halderman, and Vern Paxson. The Security Impact of HTTPS Interception. In *Proceedings of NDSS 2017*. The Internet Society, 2017.

[116] Dom Elliott. Recent Android App Bundle improvements and timeline for new apps on Google Play, 2020. `https://android-developers.googleblog.com/2020/08/recent-android-app-bundle-improvements.html`. Accessed: 2021-07-16.

[117] Dom Elliott. The future of Android App Bundles is here, 2021. `https://android-developers.googleblog.com/2021/06/the-future-of-android-app-bundles-is.html`. Accessed: 2021-07-16.

[118] Sascha Fahl, Sergej Dechand, Henning Perl, Felix Fischer, Jaromir Smrcek, and Matthew Smith. Hey, NSA: Stay Away from my Market! Future Proofing App Markets against Powerful Attackers. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 1143–1155. ACM, 2014.

[119] Valeh Farzaliyev. Code repository of the microcontroller-based voting client. https://github.com/Valeh2012/PersonalVotingMachine. Accessed: 2021-11-30.

[120] Valeh Farzaliyev, Kristjan Krips, and Jan Willemson. Developing a Personal Voting Machine for the Estonian Internet Voting System. In Chih-Cheng Hung, Jiman Hong, Alessio Bechini, and Eunjee Song, editors, *SAC '21: The 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, Republic of Korea, March 22-26, 2021*, pages 1607–1616. ACM, 2021.

[121] Ariel J. Feldman, J. Alex Halderman, and Edward W. Felten. Security Analysis of the Diebold AccuVote-TS Voting Machine. In *2007 USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'07*, 2007.

[122] Paul Feldman. A Practical Scheme for Non-interactive Verifiable Secret Sharing. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 427–437, 1987.

[123] Gabbi Fisher and Luke Valenta. Monsters in the Middleboxes: Introducing Two New Tools for Detecting HTTPS Interception. 2019. https://blog.cloudflare.com/monsters-in-the-middleboxes/. Accessed: 2021-07-16.

[124] Peter A. Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.

[125] International Institute for Democracy and Electoral Assistance. *Elections and COVID-19*. International Institute for Democracy and Electoral Assistance, 2020. Technical Paper. https://www.idea.int/sites/default/files/publications/elections-and-covid-19.pdf.

[126] OSCE Office for Democratic Institutions and Human Rights. Republic of Estonia, Parliamentary Elections, 4 March 2007: OSCE/ODIHR Election Assessment Mission Report, June 2007. https://www.osce.org/files/f/documents/1/1/25925.pdf.

[127] OSCE Office for Democratic Institutions and Human Rights. Estonia, Parliamentary Election, 6 March 2011: OSCE/ODIHR Election Assessment Mission Report, May 2011. https://www.osce.org/files/f/documents/a/9/77557.pdf.

[128] OSCE Office for Democratic Institutions and Human Rights. United Kingdom of the Great Britain and Northern Ireland, General Election, 7 May 2015: OSCE/ODIHR Election Expert Team Final Report, 2015.

[129] John C Fortier and Norman J Ornstein. The Absentee Ballot and the Secret Ballot: Challenges for Election Reform. *U. Mich. JL Reform*, 36:483–516, 2003. https://repository.law.umich.edu/mjlr/vol36/iss3/2. Accessed: 2021-03-22.

[130] Richard Frankland, Denise Demirel, Jurlind Budurushi, and Melanie Volkamer. Side-channels and eVoting machine security: Identifying vulnerabilities and defining requirements. In *2011 International Workshop on Requirements Engineering for Electronic Voting Systems, REVOTE 2011, Trento, Italy, August 29, 2011*, pages 37–46. IEEE Computer Society, 2011.

[131] Jeffrey Friedman. TEMPEST: A Signal Problem. *NSA Cryptologic Spectrum*, 2(3):26–30, 1972. `https://www.nsa.gov/Portals/70/documents/news-features/declassified-documents/cryptologic-spectrum/tempest.pdf`. Accessed: 2021-03-22.

[132] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology - AUSCRYPT '92, Workshop on the Theory and Application of Cryptographic Techniques, Gold Coast, Queensland, Australia, December 13-16, 1992, Proceedings*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992.

[133] Galois and Free & Fair. The BESSPIN Voting System, 2019. `https://github.com/GaloisInc/BESSPIN-Voting-System-Demonstrator-2019`. Accessed: 2021-07-13.

[134] Rich Geldreich. Miniz. `https://github.com/richgel999/miniz`. Accessed: 2021-11-30.

[135] Daniel Genkin, Itamar Pipman, and Eran Tromer. Get Your Hands Off My Laptop: Physical Side-Channel Key-Extraction Attacks on PCs - Extended Version. *Journal of Cryptographic Engineering*, 5(2):95–112, 2015.

[136] Kristian Gjøsteen, Clémentine Gritti, and Kelsey N Moran. Ballot Logistics: Tracking Paper-based Ballots Using Cryptography. *E-Vote-ID 2020*, pages 259–274, 2020.

[137] Kobi Glick. Enroll for app signing in the Google Play Console & secure your app using Google's robust security infrastructure, 2017. `https://android-developers.googleblog.com/2017/09/enroll-for-app-signing-in-google-play.html`. Accessed: 2021-07-16.

[138] Uwe Gnadenteich. Täna algab elektrooniline proovihääletus (in Estonian), 2011. `https://www.postimees.ee/384610/tana-algab-elektrooniline-proovihaaletus`. Accessed: 2021-10-20.

[139] Vindu Goel. Beware, iPhone Users: Fake Retail Apps Are Surging Before Holidays. 2016. `https://www.nytimes.com/2016/11/07/technology/more-iphone-fake-retail-apps-before-holidays.html`. Accessed: 2021-07-16.

[140] Ben Goldsmith. Guidelines for Trialling E-Voting in National Elections. In Feng Hao and Peter Y. A. Ryan, editors, *Real-World Electronic Voting: Design, Analysis and Deployment*, pages 19–48. CRC Press, 2017.

[141] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985.

[142] Rop Gonggrijp and Willem-Jan Hengeveld. Studying the Nedap/Groenendaal ES3B Voting Computer: A Computer Security Perspective. In Ray Martinez and David A. Wagner, editors, *2007 USENIX/ACCURATE Electronic Voting Technology Workshop, EVT'07, Boston, MA, USA, August 6, 2007*. USENIX Association, 2007.

[143] Dan Goodin. Kaspersky: Yes, we obtained NSA secrets. No, we didn't help steal them. *Ars Technica*, November 2017. `https://arstechnica.com/information-technology/2017/11/kaspersky-yes-we-obtained-nsa-secrets-no-we-didnt-help-steal-them/`. Accessed: 2021-07-13.

[144] Google. APK Signature Scheme v3. `https://source.android.com/security/apksigning/v3`. Accessed: 2021-11-30.

[145] Google. Certificate Transparency – Locally-trusted CAs. `https://chromium.googlesource.com/chromium/src/+/refs/heads/main/net/docs/certificate-transparency.md#Locally_trusted-CAs`. Accessed: 2022-03-21.

[146] Google. Code transparency for app bundles. `https://developer.android.com/guide/app-bundle/code-transparency`. Accessed: 2021-11-30.

[147] Andy Greenberg. The Untold Story of NotPetya, the Most Devastating Cyberattack in History. *Wired*, September 2018. `https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/`.

[148] Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. Fake news on Twitter during the 2016 U.S. presidential election. *Science*, 363(6425):374–378, 2019.

[149] Dimitris A Gritzalis. Principles and Requirements for a Secure E-voting System. *Computers & Security*, 21(6):539–556, 2002.

[150] Bluetooth Special Interest Group. Bluetooth Core Specification v5.2, 2019. `https://www.bluetooth.com/specifications/specs/core-specification/`. Accessed: 2021-07-13.

[151] Fredrik Gustafsson and Fredrik Gunnarsson. Positioning using time-difference of arrival measurements. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '03, Hong Kong, April 6-10, 2003*, volume 6, pages 553–556. IEEE, 2003.

[152] Rolf Haenni, Eric Dubuis, Reto E. Koenig, and Philipp Locher. CHVote: Sixteen Best Practices and Lessons Learned. In Robert Krimmer, Melanie Volkamer, Bernhard Beckert, Ralf Küsters, Oksana Kulyk, David Duenas-Cid, and Mikhel Solvak, editors, *Electronic Voting - 5th International Joint Conference, E-Vote-ID 2020, Bregenz, Austria, October 6-9, 2020, Proceedings*, volume 12455 of *Lecture Notes in Computer Science*, pages 95–111. Springer, 2020.

[153] Thomas Haines, Rajeev Goré, and Bhavesh Sharma. Did you mix me? Formally Verifying Verifiable Mix Nets in Electronic Voting. *Cryptology ePrint Archive, Report 2020/1114*, 2020. `https://eprint.iacr.org/2020/1114`.

[154] Thomas Haines, Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. How Not to Prove Your Election Outcome. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*, pages 644–660, Los Alamitos, CA, USA, May 2020. IEEE Computer Society.

[155] Thomas Haines and Ben Smyth. SoK: Surveying definitions of coercion resistance. *Cryptology ePrint Archive, Report 2019/822*, 2019. `https://ia.cr/2019/822`.

[156] Feng Hao, Matthew Nicolas Kreeger, Brian Randell, Dylan Clarke, Siamak Fayyaz Shahandashti, and Peter Hyun-Jeen Lee. Every Vote Counts: Ensuring Integrity in Large-Scale Electronic Voting. In *2014 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, EVT/WOTE '14, San Diego, CA, USA, August 18-19, 2014*. USENIX Association, 2014.

[157] Joseph P. Harris. *Election Administration in the United States*, volume 27 of *Institute for government research*. Brookings Institution, 1934. `https://www.nist.gov/itl/election-administration-united-states-1934-joseph-p-harris-phd`.

[158] James Heather and Steve A. Schneider. A Formal Framework for Modelling Coercion Resistance and Receipt Freeness. In *FM 2012: Formal Methods - 18th International Symposium, Paris, France, August 27-31, 2012. Proceedings*, pages 217–231, 2012.

[159] Sven Heiberg. IVXV online voting system repository. `https://github.com/vvk-ehk/ivxv`. Accessed: 2021-11-30.

[160] Sven Heiberg. Registration service (in Estonian), 2021. `https://github.com/vvk-ehk/ivxv/blob/master/Documentation/et/regteenus/regteenus.rst`. Accessed: 2021-10-27.

[161] Sven Heiberg, Kristjan Krips, and Jan Willemson. Planning the next steps for Estonian Internet voting. In Robert Krimmer, Melanie Volkamer, Bernhard Beckert, Ardita Driza Maurer, David Duenas-Cid, Stéphane Glondu, Iuliia Krivonosova, Oksana Kulyk, Ralf Küsters, Peter Roenne,

Beata Martin-Rozumilowicz, Mihkel Solvak, and Oliver Spycher, editors, *Fifth International Joint Conference on Electronic Voting E-Vote-ID 2020 : 6-9 October 2020: Proceedings*, pages 82–97. TalTech Press, 2020.

[162] Sven Heiberg, Kristjan Krips, and Jan Willemson. Mobile Voting – Still Too Risky? In Matthew Bernhard, Andrea Bracciali, Lewis Gudgeon, Thomas Haines, Ariah Klages-Mundt, Shin'ichiro Matsuo, Daniel Perez, Massimiliano Sala, and Sam Werner, editors, *Financial Cryptography and Data Security. FC 2021 International Workshops - CoDecFin, DeFi, VOTING, and WTSC, Virtual Event, March 5, 2021, Revised Selected Papers*, volume 12676 of *Lecture Notes in Computer Science*, pages 263–278. Springer, 2021.

[163] Sven Heiberg, Kristjan Krips, and Jan Willemson. Mobile Voting – Still Too Risky? *Cryptology ePrint Archive, Report 2021/787*, 2021. `https://eprint.iacr.org/2021/787`.

[164] Sven Heiberg, Peeter Laud, and Jan Willemson. The Application of I-Voting for Estonian Parliamentary Elections of 2011. In Aggelos Kiayias and Helger Lipmaa, editors, *E-Voting and Identity - Third International Conference, VoteID 2011, Tallinn, Estonia, September 28-30, 2011, Revised Selected Papers*, volume 7187 of *Lecture Notes in Computer Science*, pages 208–223. Springer, 2011.

[165] Sven Heiberg, Tarvi Martens, Priit Vinkel, and Jan Willemson. Improving the Verifiability of the Estonian Internet Voting Scheme. In Robert Krimmer, Melanie Volkamer, Jordi Barrat, Josh Benaloh, Nicole J. Goodman, Peter Y. A. Ryan, and Vanessa Teague, editors, *Electronic Voting - First International Joint Conference, E-Vote-ID 2016, Bregenz, Austria, October 18-21, 2016, Proceedings*, volume 10141 of *Lecture Notes in Computer Science*, pages 92–107. Springer, 2016.

[166] Sven Heiberg, Arnis Parsovs, and Jan Willemson. Log Analysis of Estonian Internet Voting 2013-2014. In Rolf Haenni, Reto E. Koenig, and Douglas Wikström, editors, *E-Voting and Identity - 5th International Conference, VoteID 2015, Bern, Switzerland, September 2-4, 2015, Proceedings*, volume 9269 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2015.

[167] Sven Heiberg, Arnis Parsovs, and Jan Willemson. Log Analysis of Estonian Internet Voting 2013-2015. *Cryptology ePrint Archive, Report 2015/1211*, 2015. `https://eprint.iacr.org/2015/1211`.

[168] Sven Heiberg and Jan Willemson. Modeling Threats of a Voting Method. In *Design, Development, and Use of Secure Electronic Voting Systems*, pages 128–148. IGI Global, 2014.

[169] Sven Heiberg and Jan Willemson. Verifiable Internet Voting in Estonia. In Robert Krimmer and Melanie Volkamer, editors, *6th International Con-*

ference on Electronic Voting: Verifying the Vote, EVOTE 2014, Lochau / Bregenz, Austria, October 29-31, 2014, pages 1–8. IEEE, 2014.

[170] Erik S. Herron. The effect of passive observation methods Azerbaijan's 2008 presidential election and 2009 referendum. *Electoral Studies*, 29(3):417–424, 2010. Special Symposium: Voters and Coalition Governments.

[171] Michael C. Herron and Daniel A. Smith. Postal delivery disruptions and the fragility of voting by mail: Lessons from Maine. *Research & Politics*, 8(1), 2021.

[172] Martin Hirt and Kazue Sako. Efficient Receipt-Free Voting Based on Homomorphic Encryption. In *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, pages 539–556, 2000.

[173] Simon Hug, Simone Wegmann, and Reto Wüest. Parliamentary Voting Procedures in Comparison. *West European Politics*, 38(5):940–968, 2015.

[174] Adafruit Industries. Adafruit TouchScreen Library. `https://github.com/adafruit/Adafruit_TouchScreen/`. Accessed: 2021-11-30.

[175] Vincenzo Iovino, Alfredo Rial, Peter B. Rønne, and Peter Y. A. Ryan. Using Selene to Verify Your Vote in JCJ. In *Financial Cryptography and Data Security*, volume 10323 of *LNCS*, pages 385–403. Springer, 2017.

[176] Markus Jakobsson. A Practical Mix. In *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, pages 448–461, 1998.

[177] Douglas W. Jones. A Brief Illustrated History of Voting, 2003. `https://homepage.divms.uiowa.edu/~jones/voting/pictures/`. Accessed: 2021-07-09.

[178] Douglas W. Jones and MacLean Hall. Technologists as Political Reformers: Lessons from the Early History of Voting Machines. In *Society for the History of Technology Annual Meeting, Las Vegas*, 2006.

[179] Hugo L. Jonker and Erik P. de Vink. Formalising Receipt-Freeness. In *Information Security, 9th International Conference, ISC 2006, Samos Island, Greece, August 30 - September 2, 2006, Proceedings*, pages 476–488, 2006.

[180] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-Resistant Electronic Elections. *Cryptology ePrint Archive, Report 2002/165*, 2002. `https://eprint.iacr.org/2002/165`.

[181] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-Resistant Electronic Elections. In *Proceedings of WPES 2005*, pages 61–70. ACM, 2005.

[182] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-Resistant Electronic Elections. In David Chaum, Markus Jakobsson, Ronald L. Rivest, Peter Y. A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections, New Directions in Electronic Voting*, volume 6000 of *Lecture Notes in Computer Science*, pages 37–63. Springer, 2010.

[183] Michael Kan. Google Researchers Find Design Flaw in Avast Antivirus. 2020. `https://www.pcmag.com/news/google-researchers-find-design-flaw-in-avast-antivirus`. Accessed: 2021-07-14.

[184] George Karantzas and Constantinos Patsakis. An Empirical Assessment of Endpoint Detection and Response Systems against Advanced Persistent Threats Attack Vectors. *Journal of Cybersecurity and Privacy*, 1(3):387–421, 2021.

[185] Shahram Khazaei and Douglas Wikström. Return Code Schemes for Electronic Voting Systems. In *Electronic Voting - Second International Joint Conference, E-Vote-ID 2017, Bregenz, Austria, October 24-27, 2017, Proceedings*, pages 198–209, 2017.

[186] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-End Verifiable Elections in the Standard Model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 468–498. Springer, 2015.

[187] Christian Killer and Burkhard Stiller. The Swiss Postal Voting Process and Its System and Security Analysis. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Bernhard Beckert, Ralf Küsters, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - 4th International Joint Conference, E-Vote-ID 2019, Bregenz, Austria, October 1-4, 2019, Proceedings*, volume 11759 of *Lecture Notes in Computer Science*, pages 134–149. Springer, 2019.

[188] Joseph Kiniry, Daniel Zimmerman, Daniel Wagner, Philip Robinson, Adam Foltzer, and Shaptar Morina. The future of voting: end-to-end verifiable Internet voting, 2015. U.S. Vote Foundation, `https://www.usvotefoundation.org/E2E-VIV`. Accessed: 2021-03-22.

[189] Gabor Kiss-Vamosi. LVGL - Light and Versatile Graphics Library. `https://github.com/lvgl/lvgl`. Accessed: 2021-11-30.

[190] Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin, and Dan S. Wallach. Analysis of an Electronic Voting System. In *2004 IEEE Symposium on Security and Privacy (S&P 2004), 9-12 May 2004, Berkeley, CA, USA*, page 27, 2004.

[191] Arne Koitmäe, Jan Willemson, and Priit Vinkel. Vote Secrecy and Voter Feedback in Remote Voting - Can We Have Both? In Robert Krimmer, Melanie Volkamer, David Duenas-Cid, Oksana Kulyk, Peter B. Rønne, Mihkel Solvak, and Micha Germann, editors, *Electronic Voting - 6th International Joint Conference, E-Vote-ID 2021, Virtual Event, October 5-8, 2021, Proceedings*, volume 12900 of *Lecture Notes in Computer Science*, pages 140–154. Springer, 2021.

[192] Oliver Kraus. U8g2: Library for monochrome displays, version 2. `https://github.com/olikraus/u8g2`. Accessed: 2021-11-30.

[193] Robert Krimmer, David Duenas-Cid, and Iuliia Krivonosova. Debate: safeguarding democracy during pandemics. Social distancing, postal, or internet voting—the good, the bad or the ugly? *Public Money & Management*, 41(1):8–10, 2021.

[194] Kristjan Krips, Ivo Kubjas, and Jan Willemson. An Internet Voting Protocol with Distributed Verification Receipt Generation. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, David Duenas-Cid, Rajeev Goré, Manik Hapsara, Reto Koenig, Steven Martin, Ronan McDermott, Peter Roenne, Uwe Serdült, and Tomasz Truderung, editors, *Third International Joint Conference on Electronic Voting E-Vote-ID 2018: 2–5 October 2018, Bregenz, Austria: Proceedings*, pages 128–146. TalTech Press, 2018.

[195] Kristjan Krips and Jan Willemson. On Practical Aspects of Coercion-Resistant Remote Voting Systems. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Bernhard Beckert, Ralf Küsters, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - 4th International Joint Conference, E-Vote-ID 2019, Bregenz, Austria, October 1-4, 2019, Proceedings*, volume 11759 of *Lecture Notes in Computer Science*, pages 216–232. Springer, 2019.

[196] Kristjan Krips, Jan Willemson, and Sebastian Värv. Implementing an Audio Side Channel for Paper Voting. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - Third International Joint Conference, E-Vote-ID 2018, Bregenz, Austria, October 2-5, 2018, Proceedings*, volume 11143 of *Lecture Notes in Computer Science*, pages 132–145. Springer, 2018.

[197] Kristjan Krips, Jan Willemson, and Sebastian Värv. Is Your Vote Overheard? A New Scalable Side-Channel Attack Against Paper Voting. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pages 621–634. IEEE, 2019.

[198] Markus G. Kuhn. Electromagnetic Eavesdropping Risks of Flat-Panel Displays. In David M. Martin Jr. and Andrei Serjantov, editors, *Privacy Enhancing Technologies, 4th International Workshop, PET 2004, Toronto,*

Canada, May 26-28, 2004, Revised Selected Papers, volume 3424 of *Lecture Notes in Computer Science*, pages 88–107. Springer, 2004.

[199] Oksana Kulyk and Stephan Neumann. Human Factors in Coercion Resistant Internet Voting–A Review of Existing Solutions and Open Challenges. In *Proceedings of the Fifth International Joint Conference on Electronic Voting E-Vote-ID 2020*, pages 189–204. TalTech Press, 2020.

[200] Oksana Kulyk, Vanessa Teague, and Melanie Volkamer. Extending Helios Towards Private Eligibility Verifiability. In *VoteID 2015, Proceedings*, volume 9269 of *LNCS*, pages 57–73. Springer, 2015.

[201] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: definition and relationship to verifiability. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010*, pages 526–535. ACM, 2010.

[202] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Verifiability, Privacy, and Coercion-Resistance: New Insights from a Case Study. In *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA*, pages 538–553. IEEE Computer Society, 2011.

[203] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Clash Attacks on the Verifiability of E-Voting Systems. In *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*, pages 395–409. IEEE Computer Society, 2012.

[204] G. N. Lance and W. T. Williams. Computer Programs for Hierarchical Polythetic Classification ("Similarity Analyses"). *The Computer Journal*, 9(1):60–64, 1966.

[205] Todd Landman and Luca Di Gennaro Splendore. Pandemic democracy: elections and COVID-19. *Journal of Risk Research*, 23(7-8):1060–1066, 2020.

[206] B. Laurie, A. Langley, and E. Kasper. Certificate Transparency. RFC 6962, RFC Editor, June 2013. `https://datatracker.ietf.org/doc/html/rfc6962`.

[207] Nate Lawson. Side-Channel Attacks on Cryptographic Software. *IEEE Security and Privacy*, 7(6):65–68, 2009.

[208] Douglas J. Leith. Web Browser Privacy: What Do Browsers Say When They Phone Home? SCSS Technical Report, 24th Feb 2020, 2020.

[209] Ronald Liive. Arvutiteadlane tegi kättesaadavaks e-valimiste koodi, mida valimisteenistus on seni kiivalt varjanud (in Estonian). *Geenius Meedia*, October 2021. `https://digi.geenius.ee/eksklusiiv/arvutiteadlane-tegi-kattesaadavaks-e-valimiste-koodi-mida-valimisteenistus-on-seni-kiivalt-varjanud/`. Accessed: 2021-11-01.

[210] Mark Lindeman. Evidence-based elections: Beyond the "rigging" debate. *Significance*, 14(1):18–23, 2017.

[211] Mark Lindeman and Philip B. Stark. A Gentle Introduction to Risk-Limiting Audits. *IEEE Secur. Priv.*, 10(5):42–49, 2012.

[212] Helger Lipmaa and Oleg Mürk. E-valimiste realiseerimisvõimaluste analüüs (in Estonian), April 2001. `https://web.archive.org/web/20040704034256fw_/http://www.vvk.ee/elektr/docs/lipmaamyrk.pdf`.

[213] Philipp Locher and Rolf Haenni. Receipt-free remote electronic elections with everlasting privacy. *Ann. des Télécommunications*, 71(7-8):323–336, 2016.

[214] Meng Luo, Pierre Laperdrix, Nima Honarmand, and Nick Nikiforakis. Time Does Not Heal All Wounds: A Longitudinal Analysis of Security-Mechanism Support in Mobile Browsers. In *Proceedings of NDSS 2019*. The Internet Society, 2019.

[215] Meng Luo, Oleksii Starov, Nima Honarmand, and Nick Nikiforakis. Hindsight: Understanding the Evolution of UI Vulnerabilities in Mobile Browsers. In *Proceedings of the 2017 ACM CCS*, CCS '17, pages 149–162. ACM, 2017.

[216] Ülle Madise and Tarvi Martens. E-voting in Estonia 2005. The first practice of country-wide binding Internet voting in the world. In Robert Krimmer, editor, *Electronic Voting 2006*, volume 86 of *LNI*, pages 15–26. GI, 2006.

[217] Ülle Madise and Priit Vinkel. *Internet Voting in Estonia: From Constitutional Debate to Evaluation of Experience over Six Elections*, pages 53–72. Springer International Publishing, Cham, 2014.

[218] Emmanouil Magkos, Mike Burmester, and Vassilios Chrissikopoulos. Receipt-Freeness in Large-Scale Elections without Untappable Channels. In *Towards The E-Society: E-Commerce, E-Business, and E-Government, The First IFIP Conference on E-Commerce, E-Business, E-Government (I3E 2001), October 3-5, Zürich, Switzerland*, pages 683–693, 2001.

[219] Richard H. McKay, Paul G. Ziebold, James D. Kirby, Douglas R. Hetzel, and James U. Snydacker. Electronic voting machine, February 1974. U.S. Patent 3793505.

[220] Microsoft. ElectionGuard, 2019. `https://github.com/microsoft/electionguard`. Accessed: 2021-03-22.

[221] Charlie Miller. Mobile Attacks and Defense. *IEEE Secur. Priv.*, 9(4):68–70, 2011.

[222] Richard Moore. QRCode library. `https://github.com/ricmoo/QRCode`. Accessed: 2021-11-30.

[223] Samuel K. Moore. Darpa Hacks Its Secure Hardware, Fends Off Most Attacks, 2021. `https://spectrum.ieee.org/tech-talk/computin`

g/embedded-systems/darpa-hacks-its-secure-hardware-fends-off-most-attacks. Accessed: 2021-07-13.

[224] Tal Moran and Moni Naor. Receipt-Free Universally-Verifiable Voting with Everlasting Privacy. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, volume 4117 of *Lecture Notes in Computer Science*, pages 373–392. Springer, 2006.

[225] James M. Moses and K. P. Trout. A Simple Laser Microphone for Classroom Demonstration. *The Physics Teacher*, 44(9):600–603, 2006.

[226] Robert Müller-Török, Arne Pautsch, and Alexander Prosser. Legal Aspects of Cross-Border Delivery of Voting Documents: a Neglected Issue? In *Proceedings of the 2015 2nd International Conference on Electronic Governance and Open Society: Challenges in Eurasia, EGOSE 2015, St. Petersburg, Russian Federation, November 24-25, 2015*, pages 123–128. ACM, 2015.

[227] Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, and Vashek Matyas. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli. In *24th ACM Conference on Computer and Communications Security (CCS'2017)*, pages 1631–1648. ACM, 2017.

[228] André Silva Neto, Matheus Leite, Roberto Araújo, Marcelle Pereira Mota, Nelson Cruz Sampaio Neto, and Jacques Traoré. Usability Considerations For Coercion-Resistant Election Systems. In *Proceedings of the 17th Brazilian Symposium on Human Factors in Computing Systems*, IHC 2018, pages 40:1–40:10, 2018.

[229] ACE Electoral Knowledge Network. Use of Cameras/Webcams in Polling Stations as Confidence Building Measures, 2014. https://aceproject.org/electoral-advice/archive/questions/replies/291099047. Accessed: 2021-11-30.

[230] Stephan Neumann, Christian Feier, Melanie Volkamer, and Reto Koenig. Towards A Practical JCJ / Civitas Implementation. *Cryptology ePrint Archive, Report 2013/464*, 2013. https://eprint.iacr.org/2013/464.

[231] Stephan Neumann and Melanie Volkamer. Civitas and the Real World: Problems and Solutions from a Practical Point of View. In *Seventh International Conference on Availability, Reliability and Security, Prague, ARES 2012, Czech Republic, August 20-24, 2012*, pages 180–185. IEEE Computer Society, 2012.

[232] Stephan Neumann, Melanie Volkamer, Jurlind Budurushi, and Marco Prandini. SecIVo: a quantitative security evaluation framework for internet voting schemes. *Ann. des Télécommunications*, 71(7-8):337–352, 2016.

[233] Lily Hay Newma. No One Knows How Deep Russia's Hacking Rampage Goes. *Wired*, December 2020. https://www.wired.com/story/russia-solarwinds-supply-chain-hack-commerce-treasury/.

[234] Lily Hay Newman. Inside the Unnerving Supply Chain Attack That Corrupted CCleaner. *Wired*, April 2018.

[235] Terry Newman. Tasmania and the Secret Ballot. *Australian Journal of Politics & History*, 49(1):93–101, 2003.

[236] Sen Nie, Ling Liu, and Yuefeng Du. Free-fall: Hacking tesla from wireless to can bus. *Briefing, Black Hat USA*, 25:1–16, 2017.

[237] Valtteri Niemi and Ari Renvall. How to Prevent Buying of Votes in Computer Elections. In *Advances in Cryptology - ASIACRYPT '94, 4th International Conference on the Theory and Applications of Cryptology, Wollongong, Australia, November 28 - December 1, 1994, Proceedings*, pages 164–170, 1994.

[238] Abdul Noury, Abel François, Olivier Gergaud, and Alexandre Garel. How does COVID-19 affect electoral participation? evidence from the French municipal elections. *PLOS ONE*, 16(2), February 2021.

[239] Supreme Court of Estonia. Constitutional judgment 3-4-1-4-11, 2011. https://www.riigikohus.ee/en/constitutional-judgment-3-4-1-4-11. Accessed: 2021-10-14.

[240] The Federal Assembly of the Swiss Confederation. Federal Act on Political Rights, Art. 14, 1976. Available at: https://www.fedlex.admin.ch/eli/cc/1978/688_688_688/en#art_14 (Accessed: 06.05.2021).

[241] Parliament of the United Kingdom. Representation of the People Act 1983, c. 2. Available at: https://www.legislation.gov.uk/ukpga/1983/2#schedule-1-paragraph-57 (Accessed: 06.05.2021).

[242] Estonian State Electoral Office. Handbook for local elections in 2017 (in Estonian). https://www.valimised.ee/sites/default/files/uploads/kov2017/KOV2017_kasiraamat_web.pdf. Accessed: 2021-03-22.

[243] Estonian State Electoral Office. General Framework of Electronic Voting and Implementation thereof at National Elections in Estonia, 2017. IVXV-ÜK-1.0, https://www.valimised.ee/sites/default/files/uploads/eng/IVXV-UK-1.0-eng.pdf.

[244] State Electoral Office. IVXV: I-voting Handbook (in Estonian), 2019. Document version IVXV-KR-0.6, https://www.valimised.ee/sites/default/files/uploads/eh/IVXV-KR-0.6.pdf. Accessed: 2021-10-28.

[245] Tatsuaki Okamoto. An electronic voting scheme. In Nobuyoshi Terashima and Edward Altman, editors, *Advanced IT Tools, IFIP World Conference on IT Tools, 2-6 September 1996, Canberra, Australia.*, pages 21–30. Chapman & Hall, 1996.

[246] Tatsuaki Okamoto. Receipt-Free Electronic Voting Schemes for Large Scale Elections. In *Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, Proceedings*, pages 25–35, 1997.

[247] Ersin Öksüzoglu and Dan S. Wallach. VoteBox Nano: A Smaller, Stronger FPGA-based Voting Machine. In *2009 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '09*. USENIX, 2009.

[248] Tavis Ormandy. Sophail: A Critical Analysis of Sophos Antivirus, 2011. Black Hat USA.

[249] Tavis Ormandy. How to Compromise the Enterprise Endpoint. 2016. `https://googleprojectzero.blogspot.com/2016/06/how-to-compromise-enterprise-endpoint.html`. Accessed: 2021-07-14.

[250] Graeme Orr. Ballot order: Donkey voting in Australia. *Election Law Journal: Rules, Politics, and Policy*, 1(4):573–578, 2002.

[251] Martin Paljak. Insecure HP USB Smart Card Keyboard, 2011. `https://web.archive.org/web/20110802021123/http://martinpaljak.net/2011/03/19/insecure-hp-usb-smart-card-keyboard/`. Accessed: 2021-07-08.

[252] Arnis Parsovs. Practical Issues with TLS Client Certificate Authentication. In *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014*. The Internet Society, 2014.

[253] Arnis Parsovs. Estonian Electronic Identity Card: Security Flaws in Key Management. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 1785–1802. USENIX Association, 2020.

[254] Arnis Parsovs. *Estonian Electronic Identity Card and its Security Challenges*. PhD thesis, University of Tartu, 2021.

[255] Stefan Patachi and Carsten Schürmann. Eos a Universal Verifiable and Coercion Resistant Voting Protocol. In Robert Krimmer, Melanie Volkamer, Nadja Braun Binder, Norbert Kersting, Olivier Pereira, and Carsten Schürmann, editors, *Electronic Voting - Second International Joint Conference, E-Vote-ID 2017, Bregenz, Austria, October 24-27, 2017, Proceedings*, volume 10615 of *Lecture Notes in Computer Science*, pages 210–227. Springer, 2017.

[256] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[257] Gordon Pennycook and David G Rand. Examining false beliefs about voter fraud in the wake of the 2020 Presidential Election. *The Harvard Kennedy School Misinformation Review*, 2(1), 2021.

[258] Olivier Pereira. Individual Verifiability and Revoting in the Estonian Internet Voting System. *Cryptology ePrint Archive, Report 2021/1098*. https://eprint.iacr.org/2021/1098.

[259] Nicole Perlroth and Scott Shane. How Israel Caught Russian Hackers Scouring the World for U.S. Secrets. *New York Times*, October 2017. https://www.nytimes.com/2017/10/10/technology/kaspersky-lab-israel-russia-hacking.html. Accessed: 2021-07-13.

[260] Sandra Petronio. *Boundaries of Privacy: Dialectics of Disclosure*. Boundaries of Privacy: Dialectics of Disclosure. State University of New York Press, 2002.

[261] Wolter Pieters. Combatting Electoral Traces: The Dutch Tempest Discussion and Beyond. In Peter Y. A. Ryan and Berry Schoenmakers, editors, *E-Voting and Identity, Second International Conference, VoteID 2009, Luxembourg, September 7-8, 2009. Proceedings*, volume 5767 of *Lecture Notes in Computer Science*, pages 172–190. Springer, 2009.

[262] Estonian Police and Border Guard. Applying for mobile-ID. https://www.politsei.ee/en/instructions/mobile-id/applying-for-mobile-id. Accessed: 2021-07-06.

[263] Estonian Police and Border Guard. ID-card sample. https://www.politsei.ee/en/instructions/id-card-sample. Accessed: 2021-07-06.

[264] Swiss Post. SwissPost Voting System architecture document. Document version 0.9.1, https://gitlab.com/swisspost-evoting/e-voting/e-voting-documentation/-/raw/master/System/SwissPost_Voting_System_architecture_document.pdf. Accessed: 2021-12-05.

[265] Swiss Post. Swiss Post Voting System, System specification, Version 0.9.6, 2021. https://gitlab.com/swisspost-evoting/documentation/-/blob/master/System/System_Specification.pdf. Accessed: 2021-06-29.

[266] Jordi Puiggali, Jordi Cucurull, Sandra Guasch, and Robert Krimmer. Verifiability Experiences in Government Online Voting Systems. In Robert Krimmer, Melanie Volkamer, Nadja Braun Binder, Norbert Kersting, Olivier Pereira, and Carsten Schürmann, editors, *Electronic Voting - Second International Joint Conference, E-Vote-ID 2017, Bregenz, Austria, October 24-27, 2017, Proceedings*, volume 10615 of *Lecture Notes in Computer Science*, pages 248–263. Springer, 2017.

[267] Matt Qvortrup. First past the Postman: Voting by Mail in Comparative Perspective. *The Political Quarterly*, 76(3):414–419, 2005.

[268] Jathushan Rajasegaran, Naveen Karunanayake, Ashanie Gunathillake, Suranga Seneviratne, and Guillaume Jourjon. A Multi-modal Neural Embeddings Approach for Detecting Mobile Counterfeit Apps. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3165–3171. ACM, 2019.

[269] Brian Randell and Peter Y. A. Ryan. Voting Technologies and Trust. *IEEE Security & Privacy*, 4(5):50–56, 2006.

[270] Keith Rebello. System Security Integration Through Hardware and Firmware (SSITH). DARPA research program. `https://www.darpa.mil/program/ssith`.

[271] S. Reis, V. Correia, M. Martins, G. Barbosa, R. M. Sousa, G. Minas, S. Lanceros-Mendez, and J. G. Rocha. Touchscreen based on acoustic pulse recognition with piezoelectric polymer sensors. In *2010 IEEE International Symposium on Industrial Electronics*, pages 516–520, 2010.

[272] Andrew Reynolds and Marco Steenbergen. How the World Votes: The Political Consequences of Ballot Design, Innovation and Manipulation. *Electoral Studies*, 25(3):570–598, 2006.

[273] Allen Thorndike Rice. The Next National Reform. *The North American Review*, 148(386):82–85, 1889.

[274] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to Leak a Secret. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[275] Ronald L. Rivest and Madars Virza. Software Independence Revisited. In Feng Hao and Peter Y. A. Ryan, editors, *Real-World Electronic Voting: Design, Analysis and Deployment*, pages 3–17. CRC Press, 2017.

[276] Kathryn Rix. The Second Reform Act and the Problem of Electoral Corruption. *Parliamentary History*, 36(1):64–81, 2017.

[277] James Robert. Pydub, 2011. `https://github.com/jiaaro/pydub`. Accessed: 2021-09-22.

[278] Rovetta, D. and Sarti, A. and Tubaro, S. and Colombo, G. Modelling elastic wave propagation in thin plates. In D.T. Pham, E.E. Eldukhri, and A.J. Soroka, editors, *Intelligent Production Machines and Systems*, pages 548–555. Elsevier Science Ltd, 2006.

[279] Peter Y. A. Ryan, Peter B. Rønne, and Vincenzo Iovino. Selene: Voting with Transparent Verifiability and Coercion-Mitigation. *Cryptology ePrint Archive, Report 2015/1105*, 2015. `https://eprint.iacr.org/2015/1105`.

[280] Peter Y. A. Ryan, Peter B. Rønne, and Vincenzo Iovino. Selene: Voting with Transparent Verifiability and Coercion-Mitigation. In *FC 2016 International Workshops, Revised Selected Papers*, volume 9604 of *LNCS*, pages 176–192. Springer, 2016.

[281] Peter Y. A. Ryan and Steve A. Schneider. Prêt à Voter with Re-encryption Mixes. In Dieter Gollmann, Jan Meier, and Andrei Sabelfeld, editors, *Computer Security - ESORICS 2006, 11th European Symposium on Research in Computer Security, Hamburg, Germany, September 18-20, 2006, Proceedings*, volume 4189 of *Lecture Notes in Computer Science*, pages 313–326. Springer, 2006.

[282] Kazue Sako and Joe Kilian. Receipt-Free Mix-Type Voting Scheme - A Practical Solution to the Implementation of a Voting Booth. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology - EUROCRYPT '95, International Conference on the Theory and Application of Cryptographic Techniques, Saint-Malo, France, May 21-25, 1995, Proceeding*, volume 921 of *Lecture Notes in Computer Science*, pages 393–403. Springer, 1995.

[283] Krishna Sampigethaya and Radha Poovendran. A framework and taxonomy for comparison of electronic voting schemes. *Computers & Security*, 25(2):137–153, 2006.

[284] Frederic Charles Schaffer. *Elections for Sale: The Causes and Consequences of Vote Buying*. Governance and Political Change Series. Ateneo De Manila University Press, 2007.

[285] Hidenori Sekiguchi. Information leakage of input operation on touch screen monitors caused by electromagnetic noise. In *2010 IEEE International Symposium on Electromagnetic Compatibility*, pages 127–131. IEEE, 2010.

[286] Adi Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.

[287] Fatemeh Shirazi, Stephan Neumann, Ines Ciolacu, and Melanie Volkamer. Robust electronic voting: Introducing robustness in Civitas. In *2011 International Workshop on Requirements Engineering for Electronic Voting Systems, REVOTE 2011, Trento, Italy, August 29, 2011*, pages 47–55. IEEE Computer Society, 2011.

[288] Ilia Shumailov, Laurent Simon, Jeff Yan, and Ross Anderson. Hearing your touch: A new acoustic side channel on smartphones. *arXiv preprint arXiv:1903.11137*, 2019. `https://arxiv.org/abs/1903.11137`.

[289] Peter Smulders. The Threat of Information Theft by Reception of Electromagnetic Radiation from RS-232 Cables. *Computers & Security*, 9(1):53–58, 1990.

[290] Mihkel Solvak, Taavi Unt, Dmitri Rozgonjuk, Andres Võrk, Märten Veskimäe, and Kristjan Vassil. E-governance diffusion: Population level

e-service adoption rates and usage patterns. *Telematics Informatics*, 36:39–54, 2019.

[291] Michael A. Specter, James Koppel, and Daniel Weitzner. The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz, the First Internet Voting Application Used in U.S. Federal Elections. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1535–1553. USENIX Association, August 2020.

[292] M.A. Spirito and A.G. Mattioli. On the hyperbolic positioning of GSM mobile stations. In *1998 URSI International Symposium on Signals, Systems, and Electronics. Conference Proceedings (Cat. No.98EX167)*, pages 173–177, 1998.

[293] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J. Alex Halderman. Security Analysis of the Estonian Internet Voting System. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.

[294] Isabelle Stadelmann-Steffen and Marlène Gerber. Voting in the rain: The impact of rain on participation in open-air assemblies. *Local Government Studies*, 46(3):414–435, 2020.

[295] Matthias Stadler. Swiss Abroad furious about missing voting documents, 2020. `https://www.swissinfo.ch/eng/swiss-abroad-furious-about-missing-voting-documents/46057748`. Accessed: 2021-10-14.

[296] Ida Sofie Gebhardt Stenerud and Christian Bull. When Reality Comes Knocking Norwegian Experiences with Verifiable Electronic Voting. In *5th International Conference on Electronic Voting 2012, (EVOTE 2012), Co-organized by the Council of Europe, Gesellschaft für Informatik and E-Voting.CC, July 11-14, 2012, Castle Hofen, Bregenz, Austria*, pages 21–33, 2012.

[297] Orathai Sukwong, Hyong S. Kim, and James C. Hoe. Commercial Antivirus Software Effectiveness: An Empirical Study. *Computer*, 44(3):63–70, 2011.

[298] Espressif Systems. Arduino core for the ESP32, ESP32-S2 and ESP32-C3. `https://github.com/espressif/arduino-esp32`. Accessed: 2021-11-30.

[299] Espressif Systems. Customized Setup of Toolchain. `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/toolchain-setup-scratch.html`. Accessed: 2021-11-30.

[300] Espressif Systems. ESP32. `https://www.espressif.com/en/products/socs/esp32`. Accessed: 2021-11-30.

[301] Espressif Systems. Espressif IoT Development Framework. `https://github.com/espressif/esp-idf`. Accessed: 2021-11-30.

[302] Espressif Systems. ESP32 Series Datasheet Version 3.6, 2021. https://github.com/NationalSecurityAgency/ghidra. Accessed: 2021-07-13.

[303] Mattias Tammet. Ainuke e-hääle rikkuja: Eestis ei julgeta e-hääletamist kritiseerida (in Estonian), March 2015. https://www.ohtuleht.ee/664725/ainuke-e-haale-rikkuja-eestis-ei-julgeta-e-haaletamist-kritiseerida. Accessed: 2021-10-05.

[304] Tanel Tammet and Hannu Krosing. E-valimised Eesti Vabariigis: võimaluste analüüs (in Estonian), October 2001. https://web.archive.org/web/20040704034256fw_/http://www.vvk.ee/elektr/docs/evalimisteanalyys24okt.doc.

[305] Tom Theuns. Jeremy Bentham, John Stuart Mill and the Secret Ballot: Insights from Nineteenth Century Democratic Theory. *Australian Journal of Politics & History*, 63(4):493–507, 2017.

[306] Jurij Toplak. Preferential Voting: Definition and Classification. *Lex localis - Journal of Local Self-Government*, 15(4):737–761, October 2017.

[307] Ehsan Toreini, Siamak F. Shahandashti, and Feng Hao. Texture to the Rescue: Practical Paper Fingerprinting Based on Texture Patterns. *ACM Trans. Priv. Secur.*, 20(3):9:1–9:29, 2017.

[308] Valentyna Tsap, Silvia Lips, and Dirk Draheim. Analyzing eID Public Acceptance and User Preferences for Current Authentication Options in Estonia. In Andrea Kő, Enrico Francesconi, Gabriele Kotsis, A Min Tjoa, and Ismail Khalil, editors, *Electronic Government and the Information Systems Perspective - 9th International Conference, EGOVIS 2020, Bratislava, Slovakia, September 14-17, 2020, Proceedings*, volume 12394 of *Lecture Notes in Computer Science*, pages 159–173. Springer, 2020.

[309] United Nations, Department of Economic and Social Affairs, Population Division. International Migration Report 2015, Highlights, 2016. (ST/ESA/SER.A/375), https://www.un.org/en/development/desa/population/migration/publications/migrationreport/docs/MigrationReport2015_Highlights.pdf.

[310] Dominique Unruh. Everlasting Multi-party Computation. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 380–397. Springer, 2013.

[311] Helene Unterwieser and Gerhard Schickhofer. Influence of moisture content of wood on sound velocity and dynamic MOE of natural frequency- and ultrasonic runtime measurement. *European Journal of Wood and Wood Products*, 69(2):171–181, February 2010.

[312] W. Verhelst and M. Roelands. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of

speech. In *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 554–557 vol.2, April 1993.

[313] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[314] Melanie Volkamer. *Evaluation of Electronic Voting - Requirements and Evaluation Procedures to Support Responsible Election Authorities*, volume 30 of *Lecture Notes in Business Information Processing*. Springer, 2009.

[315] Melanie Volkamer, Jurlind Budurushi, and Denise Demirel. Vote casting device with VV-SV-PAT for elections with complicated ballot papers. In *2011 International Workshop on Requirements Engineering for Electronic Voting Systems, REVOTE 2011, Trento, Italy, August 29, 2011*, pages 1–8. IEEE, 2011.

[316] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.

[317] Martin Vuagnoux and Sylvain Pasini. Compromising Electromagnetic Emanations of Wired and Wireless Keyboards. In Fabian Monrose, editor, *18th USENIX Security Symposium, Montreal, Canada, August 10-14, 2009, Proceedings*, pages 1–16. USENIX Association, 2009.

[318] Andrew Whyte and Helen Wright. Hacker downloads close to 300,000 personal ID photos. *ERR News*, June 2021. `https://news.err.ee/1608291072/hacker-downloads-close-to-300-000-personal-id-photos`. Accessed: 2021-10-27.

[319] Douglas Wikström. A Sender Verifiable Mix-Net and a New Proof of a Shuffle. In Bimal K. Roy, editor, *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, volume 3788 of *Lecture Notes in Computer Science*, pages 273–292. Springer, 2005.

[320] Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. Security Analysis of India's Electronic Voting Machines. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *Proceedings of the*

17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010, pages 1–14. ACM, 2010.

[321] Helen Wright. Election Committee: Referendum cannot be held in April. *ERR News*, November 2020. `https://news.err.ee/1157740/election-committee-referendum-cannot-be-held-in-april`. Accessed: 2021-10-27.

[322] Helen Wright. RIA: Mobile voting could be launched in 2021. *ERR News*, April 2020. `https://news.err.ee/1082021/ria-mobile-voting-could-be-launched-in-2021`. Accessed: 2021-07-15.

[323] Kanae Yoshida, Hironori Imai, Nana Serizawa, Tatsuya Mori, and Akira Kanaoka. Understanding the Origins of Weak Cryptographic Algorithms Used for Signing Android Apps. *J. Inf. Process.*, 27:593–602, 2019.

[324] Tuo Yu, Haiming Jin, and Klara Nahrstedt. WritingHacker: Audio Based Eavesdropping of Handwriting via Mobile Devices. In Paul Lukowicz, Antonio Krüger, Andreas Bulling, Youn-Kyung Lim, and Shwetak N. Patel, editors, *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016*, pages 463–473. ACM, 2016.

[325] Li Zhuang, Feng Zhou, and J. D. Tygar. Keyboard Acoustic Emanations Revisited. In Vijay Atluri, Catherine A. Meadows, and Ari Juels, editors, *Proceedings of the 12th ACM Conference on Computer and Communications Security, CCS 2005, Alexandria, VA, USA, November 7-11, 2005*, pages 373–382. ACM, 2005.

LIST OF ABBREVIATIONS

ADC	analog-to-digital converter
API	Application Programming Interface
APK	Android Package
AV	alternative vote
CT	Certificate Transparency
DARPA	Defense Advanced Research Projects Agency
DKIM	DomainKeys Identified Mail
DRE	Direct Recording Electronic voting machine
E2E	End-to-End
FFT	fast Fourier transform
FPGA	field programmable gate array
HPKP	HTTP Public Key Pinning
HSM	hardware security module
IP	Internet Protocol
IRV	instant-runoff voting
I2C	Inter-Integrated Circuit
JCJ	a voting scheme designed by Juels, Catalano, and Jakobsson
k-NN	*k*-nearest neighbors algorithm, an algorithm used for classification
MLP	Multi-layer Perceptron classifier algorithm
NEC	National Electoral Committee
NFC	Near-Field Communication
NIST	The National Institute of Standards and Technology
NSA	The National Security Agency
OCSP	Online Certificate Status Protocol
ODIHR	Office for Democratic Institutions and Human Rights
OLED	an organic light-emitting diode
OSCE	The Organization for Security and Co-operation in Europe
PC	personal computer
PET	plaintext equivalence test
PIN	personal identification number
PKI	public key infrastructure
RSA	a public-key cryptosystem created by Ron Rivest, Adi Shamir and Leonard Adleman
SIM	subscriber identity module

SPI	serial peripheral interface
SRI	Subresource Integrity
SSITH	System Security Integration Through Hardware and Firmware
STV	single transferable vote
TDOA	time difference of arrival
TLS	Transport Layer Security
Tor	The Onion Router
URL	Uniform Resource Locator
VPN	virtual private network
WSOLA	Waveform Similarity Based Overlap-Add algorithm

ACKNOWLEDGEMENTS

The research that led to this thesis was supported by the European Regional Development Fund through the Estonian Centre of Excellence in ICT Research (EXCITE) and by the Estonian Research Council through Personal research funding Team grant (PRG) number PRG920.

During my PhD studies, I was also supported by the European Social Fund through the IT Academy programme, by the Estonian Research Council's through Institutional Research Funding (IUT) grant number IUT2-1, and by the Estonian Science Foundation through Research Grant (ETF) number ETF9171.

I am very grateful for my supervisors, Jan Willemson and Sven Laur. Sven Laur introduced me to cryptography, and Jan Willemson offered me the opportunity to do research in the field of voting technologies.

I thank my co-authors Jan Willemson, Sven Heiberg, Sebastian Värv, Ivo Kubjas and Valeh Farzaliyev, with whom it has been a pleasure to work with. I also thank my colleagues from Cybernetica, who supported my research and volunteered to participate in the experiments. I am also grateful to the supportive staff from the Institute of Computer Science.

I thank the pre-reviewers Arnis Parsovs, Olivier Pereira and Carsten Schürmann for their comments and feedback. In addition, I would like to thank Anto Veldre, Priit Vinkel and Toomas Krips for reading the thesis draft and giving suggestions for improvements.

Finally, I would like to express my gratitude to my family, especially to Kadri, who supported me throughout my studies. The long writing process would not have been the same without my daughter Pärli who brought joy to each day. Thank you!

SISUKOKKUVÕTE

Valimiste privaatsus ja mõjutuskindlus

Demokraatia lahutamatuks osaks on inimeste õigus osaleda valimistel ning seeläbi määrata endale esindajad. Kuidas täpsemalt esindajad valitakse sõltub nii valimissüsteemist kui ka hääletamise korrast. Demokraatia toimimiseks on vaja tagada vabade ja ausate valimiste läbiviimine. Seetõttu on valimiste turvamine kriitilise tähtsusega. Selle eesmärgi täitmiseks tuleb täpselt aru saada kehtivatest turvanõuetest, hääletamissüsteemide arhitektuurist ning võimalikest turvanõrkustest. Olukorra muudavad keerukamaks valimistele rakendatavad vastuolulised turvanõuded, mis samaaegselt nõuavad nii hääletamise salajasust kui valimistega seotud protsesside auditeeritavust.

Seoses elektrooniliste hääletamismasinate kasutuselevõtuga kasvas valimissüsteemide auditeeritavuse ja läbipaistvuse osatähtsus. Samad põhimõtted kandusid üle ka internetihääletamissüsteemidele. Kuid kaughääletamine pole samaväärne traditsioonilise valimisjaoskonnas toimuva hääletamisprotsessiga. Erinevalt kaughääletamisest on valimisjaoskonnas hääletav valija ümbritsetud privaatsust tagava valimiskabiiniga. Seetõttu rakendatakse internetihääletamise korral meetmeid nii mõjutuskindluse suurendamiseks kui ka valimiste tervikluse tagamiseks.

Käesolev töö kirjeldab hääletamise privaatsuse ja mõjutuskindluse saavutamiseks vajalikke meetmeid ning analüüsib nende praktilist rakendatavust. Pikemalt keskendutakse tänapäevaste internetihääletusprotokollide mõjutuskindluse uurimisele. Kuna turvanõuete kohaselt on enamasti vaja tagada nii hääle kontrollitavus kui valija mõjutamatus, tuleb leida optimaalne tasakaal nende omaduste vahel. Tänapäevaste internetihääletusprotokollide analüüsimise tulemusena selgub, et hääle kontrollitavuse ja valija mõjutamatuse samaaegseks saavutamiseks on vaja aluseks võtta mitmeid eeldusi, mida on praktikas raske täita. Seetõttu tuleb teha järeleandmisi kas valimissüsteemi terviklusgarantiide või mõjutuskindluse osas.

Võrreldes internetihääletussüsteemidega on traditsiooniliste paberhääletussüsteemide turvalisust tänapäevase tehnoloogia kontekstis oluliselt vähem uuritud. Heaks näiteks on valimiskabiin, mis peaks tagama hääle privaatsuse ja valija mõjutamatuse. Samas on selge, et valijad saavad nutiseadmete abil salvestada hääletamisprotseduuri ning seeläbi tõestada mõjutajale, kelle poolt hääletati. Kui valijal on lihtne tõestada kuidas ta hääletas, siis muutub võimalikuks ka häälte müümine. Mõlemal juhul rikutaks valimiste salajasuse põhimõtet. Lisaks eelnevalt mainitule oleks tänapäevase tehnoloogia abil teoreetiliselt võimalik valimissedelitelt tuvastada ka valijate sõrmejälgi.

Siiani polnud uuritud, kas analoogselt elektroonilistele süsteemidele leidub ka paberhääletamise vastu suunatud külgkanaliründe id. Väitekirjas kirjeldame paberhääletamise protsessist leitud audiokõrvalkanalit, mis lekitab infot valija poolt tehtud valiku osas. Leiu illustreerimiseks ja tõendamiseks ehitasime kaks proto-

tüüpi, mis võimaldasid sedeli täitmisel tekkiva heli põhjal tuvastada, mis numbrid sedelile kirjutati ja kuhu sedelile märge tehti. Eelnevalt kirjeldatud näidete põhjal saab väita, et traditsiooniline paberhääletamine ei garanteeri hääle absoluutset privaatsust ega valija mõjutamatust. Seeläbi tekib uus võrdlusbaas kaughääletamissüsteemidele seatavate turvanõuete osas.

Sarnaselt teistele valimissüsteemidele oli ka Eestis kasutusel olevas internetihääletussüsteemis vaja leida tasakaal mõjutuskindluse ja terviklusomaduste vahel. Valija mõjutamatuse suurendamiseks on Eesti süsteemis kasutusel korduvhääletamise võimalus. Tervikluse tagamiseks on valijal võimalik verifitseerimisrakenduse abil kontrollida, et tema hääl jõudis kohale. Kuna serveri poolel tehtavatest toimingutest jäävad järele krüptograafilised tõestused, on audiitoritel võimalik kontrollida, et kõik kehtivad hääled loeti kokku ning hääli ei lisatud, muudetud ega eemaldatud. Siiski leidub ka Eesti internetihääletussüsteemis mitmeid nõrkusi, mis tulenevad otseselt turvanõuetes olevatest vastuoludest. Doktoritöö ühe osana kirjeldame hääletamisprotseduuri ja verifitseerimisprotseduuriga seotud nõrkusi ning pakume välja meetodeid tuvastatud probleemide lahendamiseks.

Üheks keerukamaks lahendamist vajavaks probleemiks on valijate poolt kasutatavate arvutite usaldatavuse puudumine. Kahjurvaraga kaasnevad riskid mõjutavad nii valija poolt antud hääle terviklust kui privaatsust. Kahjuks pole sellele probleemile lihtsat lahendust.

Kui hääle tervikluse tagamiseks saab lisada verifitseerimismeetodeid ning teavitusi, siis hääle privaatsust on keerukam kaitsta. Üheks võimalikuks lahenduseks oleks valikute kodeerimine nii, et arvutis olev kahjurvara ei mõistaks, kelle poolt hääl antakse. Paraku ei ole koodhääletamine kasutajasõbralik ning nõuaks terve hääletussüsteemi ümberehitamist. Alternatiivseks lahenduseks on turvaliste hääletamisseadmete kasutamine. Niisuguse lähenemise katsetamiseks töötasime välja mikrokontrolleril põhineva personaalse hääletamisseadme prototüübi. Sellise seadme kasutamine võimaldaks teadlikul valijal vähendada kahjurvara poolse sekkumise tõenäosust. Lisaks seadme kirjeldusele publitseerisime ka turvaanalüüsi ja seadme tööks vajaliku tarkvara lähtekoodi.

Viimase aspektina analüüsisime mobiilihääletamisega seonduvaid riske. Nutitelefonidel põhineva hääletamisrakenduse ehitamiseks on kaks alternatiivi. Esimeseks võimaluseks on tugineda eraldiseisvale hääletamisrakendusele, mille valija peaks ametlikust rakendustepoest alla laadima. Teiseks variandiks on kasutada brauseripõhist lähenemist, mis tähendaks hääletamisrakenduse esitamist dünaamilise veebilehena. Analüüsi tulemusena leidsime, et mõlema alternatiiviga kaasnevad turvariskid. Brauseripõhine lahendus eristus täiendavate riskide poolest, mis paljuski tulenesid mobiilsete brauseritega seotud probleemidest. Seetõttu tuleks enne mobiilhääletamise kasutuselevõttu maandada riskid, mis tulenevad mobiilseadmete riistvaralistest ja tarkvaralistest omapäradest.

CURRICULUM VITAE

Personal data

Name:	Kristjan Krips
Birth:	1987-05-18
Citizenship:	Estonian
Languages:	English, Estonian, German
E-mail:	kristjan.krips@eesti.ee

Education

2012–	University of Tartu, Ph.D. candidate in Computer Science
2010–2012	University of Tartu, M.Sc in Computer Science
2006–2010	University of Tartu, B.Sc in Computer Science
2003–2006	Hugo Treffner Gymnasium

Employment

2017–	Cybernetica AS, Junior Researcher
2016–	University of Tartu, Assistant of Informatics
2013–2016	University of Tartu, Assistant of Computer Science
2012–2013	University of Tartu, Assistant of Informatics

Scientific work

Main fields of interest:

- Voting / e-voting
- Information security
- Formal verification

ELULOOKIRJELDUS

Isikuandmed

Nimi:	Kristjan Krips
Sünniaeg ja -koht:	1987-05-18
Kodakondsus:	eestlane
Keelteoskus:	eesti, inglise, saksa
E-post:	kristjan.krips@eesti.ee

Haridus

2012–	Tartu Ülikool, informaatika doktorant
2010–2012	Tartu Ülikool, MSc informaatikas
2006–2010	Tartu Ülikool, BSc informaatikas
2003–2006	Hugo Treffneri Gümnaasium

Teenistuskäik

2017–	Cybernetica AS, nooremteadur
2016–	Tartu Ülikool, informaatika assistent
2013–2016	Tartu Ülikool, arvutisüsteemide assistent
2012–2013	Tartu Ülikool, informaatika assistent

Teadustegevus

Peamised uurimisvaldkonnad:

- Hääletamine / e-hääletamine
- Infoturve
- Formaalne verifitseerimine

LIST OF ORIGINAL PUBLICATIONS

1. Kristjan Krips, Ivo Kubjas, and Jan Willemson. An Internet Voting Protocol with Distributed Verification Receipt Generation. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, David Duenas-Cid, Rajeev Goré, Manik Hapsara, Reto Koenig, Steven Martin, Ronan McDermott, Peter Roenne, Uwe Serdült, and Tomasz Truderung, editors, *Third International Joint Conference on Electronic Voting E-Vote-ID 2018: 2–5 October 2018, Bregenz, Austria: Proceedings*, pages 128–146. TalTech Press, 2018.

2. Kristjan Krips, Jan Willemson, and Sebastian Värv. Implementing an Audio Side Channel for Paper Voting. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - Third International Joint Conference, E-Vote-ID 2018, Bregenz, Austria, October 2-5, 2018, Proceedings*, volume 11143 of *Lecture Notes in Computer Science*, pages 132–145. Springer, 2018.

3. Kristjan Krips, Jan Willemson, and Sebastian Värv. Is Your Vote Overheard? A New Scalable Side-Channel Attack Against Paper Voting. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pages 621–634. IEEE, 2019.

4. Kristjan Krips and Jan Willemson. On Practical Aspects of Coercion-Resistant Remote Voting Systems. In Robert Krimmer, Melanie Volkamer, Véronique Cortier, Bernhard Beckert, Ralf Küsters, Uwe Serdült, and David Duenas-Cid, editors, *Electronic Voting - 4th International Joint Conference, E-Vote-ID 2019, Bregenz, Austria, October 1-4, 2019, Proceedings*, volume 11759 of *Lecture Notes in Computer Science*, pages 216–232. Springer, 2019.

5. Sven Heiberg, Kristjan Krips, and Jan Willemson. Planning the next steps for Estonian Internet voting. In Robert Krimmer, Melanie Volkamer, Bernhard Beckert, Ardita Driza Maurer, David Duenas-Cid, Stéphane Glondu, Iuliia Krivonosova, Oksana Kulyk, Ralf Küsters, Peter Roenne, Beata Martin-Rozumilowicz, Mihkel Solvak, and Oliver Spycher, editors, *Fifth International Joint Conference on Electronic Voting E-Vote-ID 2020 : 6-9 October 2020: Proceedings*, pages 82–97. TalTech Press, 2020.

6. Sven Heiberg, Kristjan Krips, and Jan Willemson. Mobile Voting – Still Too Risky? In Matthew Bernhard, Andrea Bracciali, Lewis Gudgeon, Thomas Haines, Ariah Klages-Mundt, Shin'ichiro Matsuo, Daniel Perez, Massim-

iliano Sala, and Sam Werner, editors, *Financial Cryptography and Data Security. FC 2021 International Workshops - CoDecFin, DeFi, VOTING, and WTSC, Virtual Event, March 5, 2021, Revised Selected Papers*, volume 12676 of *Lecture Notes in Computer Science*, pages 263–278. Springer, 2021.

7. Valeh Farzaliyev, Kristjan Krips, and Jan Willemson. Developing a Personal Voting Machine for the Estonian Internet Voting System. In Chih-Cheng Hung, Jiman Hong, Alessio Bechini, and Eunjee Song, editors, *SAC '21: The 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, Republic of Korea, March 22-26, 2021*, pages 1607–1616. ACM, 2021.

DISSERTATIONES INFORMATICAE PREVIOUSLY PUBLISHED IN DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS

19. **Helger Lipmaa.** Secure and efficient time-stamping systems. Tartu, 1999, 56 p.
22. **Kaili Müürisep.** Eesti keele arvutigrammatika: süntaks. Tartu, 2000, 107 lk.
23. **Varmo Vene.** Categorical programming with inductive and coinductive types. Tartu, 2000, 116 p.
24. **Olga Sokratova.** Ω-rings, their flat and projective acts with some applications. Tartu, 2000, 120 p.
27. **Tiina Puolakainen.** Eesti keele arvutigrammatika: morfoloogiline ühestamine. Tartu, 2001, 138 lk.
29. **Jan Villemson.** Size-efficient interval time stamps. Tartu, 2002, 82 p.
45. **Kristo Heero.** Path planning and learning strategies for mobile robots in dynamic partially unknown environments. Tartu 2006, 123 p.
49. **Härmel Nestra.** Iteratively defined transfinite trace semantics and program slicing with respect to them. Tartu 2006, 116 p.
53. **Marina Issakova.** Solving of linear equations, linear inequalities and systems of linear equations in interactive learning environment. Tartu 2007, 170 p.
55. **Kaarel Kaljurand.** Attempto controlled English as a Semantic Web language. Tartu 2007, 162 p.
56. **Mart Anton.** Mechanical modeling of IPMC actuators at large deformations. Tartu 2008, 123 p.
59. **Reimo Palm.** Numerical Comparison of Regularization Algorithms for Solving Ill-Posed Problems. Tartu 2010, 105 p.
61. **Jüri Reimand.** Functional analysis of gene lists, networks and regulatory systems. Tartu 2010, 153 p.
62. **Ahti Peder.** Superpositional Graphs and Finding the Description of Structure by Counting Method. Tartu 2010, 87 p.
64. **Vesal Vojdani.** Static Data Race Analysis of Heap-Manipulating C Programs. Tartu 2010, 137 p.
66. **Mark Fišel.** Optimizing Statistical Machine Translation via Input Modification. Tartu 2011, 104 p.
67. **Margus Niitsoo**. Black-box Oracle Separation Techniques with Applications in Time-stamping. Tartu 2011, 174 p.
71. **Siim Karus.** Maintainability of XML Transformations. Tartu 2011, 142 p.
72. **Margus Treumuth.** A Framework for Asynchronous Dialogue Systems: Concepts, Issues and Design Aspects. Tartu 2011, 95 p.
73. **Dmitri Lepp.** Solving simplification problems in the domain of exponents, monomials and polynomials in interactive learning environment T-algebra. Tartu 2011, 202 p.

74. **Meelis Kull.** Statistical enrichment analysis in algorithms for studying gene regulation. Tartu 2011, 151 p.
77. **Bingsheng Zhang.** Efficient cryptographic protocols for secure and private remote databases. Tartu 2011, 206 p.
78. **Reina Uba.** Merging business process models. Tartu 2011, 166 p.
79. **Uuno Puus.** Structural performance as a success factor in software development projects – Estonian experience. Tartu 2012, 106 p.
81. **Georg Singer.** Web search engines and complex information needs. Tartu 2012, 218 p.
83. **Dan Bogdanov.** Sharemind: programmable secure computations with practical applications. Tartu 2013, 191 p.
84. **Jevgeni Kabanov.** Towards a more productive Java EE ecosystem. Tartu 2013, 151 p.
87. **Margus Freudenthal.** Simpl: A toolkit for Domain-Specific Language development in enterprise information systems. Tartu, 2013, 151 p.
90. **Raivo Kolde.** Methods for re-using public gene expression data. Tartu, 2014, 121 p.
91. **Vladimir Šor.** Statistical Approach for Memory Leak Detection in Java Applications. Tartu, 2014, 155 p.
92. **Naved Ahmed.** Deriving Security Requirements from Business Process Models. Tartu, 2014, 171 p.
94. **Liina Kamm.** Privacy-preserving statistical analysis using secure multi-party computation. Tartu, 2015, 201 p.
100. **Abel Armas Cervantes.** Diagnosing Behavioral Differences between Business Process Models. Tartu, 2015, 193 p.
101. **Fredrik Milani.** On Sub-Processes, Process Variation and their Interplay: An Integrated Divide-and-Conquer Method for Modeling Business Processes with Variation. Tartu, 2015, 164 p.
102. **Huber Raul Flores Macario.** Service-Oriented and Evidence-aware Mobile Cloud Computing. Tartu, 2015, 163 p.
103. **Tauno Metsalu.** Statistical analysis of multivariate data in bioinformatics. Tartu, 2016, 197 p.
104. **Riivo Talviste.** Applying Secure Multi-party Computation in Practice. Tartu, 2016, 144 p.
108. **Siim Orasmaa.** Explorations of the Problem of Broad-coverage and General Domain Event Analysis: The Estonian Experience. Tartu, 2016, 186 p.
109. **Prastudy Mungkas Fauzi.** Efficient Non-interactive Zero-knowledge Protocols in the CRS Model. Tartu, 2017, 193 p.
110. **Pelle Jakovits.** Adapting Scientific Computing Algorithms to Distributed Computing Frameworks. Tartu, 2017, 168 p.
111. **Anna Leontjeva.** Using Generative Models to Combine Static and Sequential Features for Classification. Tartu, 2017, 167 p.
112. **Mozhgan Pourmoradnasseri.** Some Problems Related to Extensions of Polytopes. Tartu, 2017, 168 p.

113. **Jaak Randmets.** Programming Languages for Secure Multi-party Computation Application Development. Tartu, 2017, 172 p.
114. **Alisa Pankova.** Efficient Multiparty Computation Secure against Covert and Active Adversaries. Tartu, 2017, 316 p.
116. **Toomas Saarsen.** On the Structure and Use of Process Models and Their Interplay. Tartu, 2017, 123 p.
121. **Kristjan Korjus.** Analyzing EEG Data and Improving Data Partitioning for Machine Learning Algorithms. Tartu, 2017, 106 p.
122. **Eno Tõnisson.** Differences between Expected Answers and the Answers Offered by Computer Algebra Systems to School Mathematics Equations. Tartu, 2017, 195 p.

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

1. **Abdullah Makkeh**. Applications of Optimization in Some Complex Systems. Tartu 2018, 179 p.
2. **Riivo Kikas**. Analysis of Issue and Dependency Management in Open-Source Software Projects. Tartu 2018, 115 p.
3. **Ehsan Ebrahimi**. Post-Quantum Security in the Presence of Superposition Queries. Tartu 2018, 200 p.
4. **Ilya Verenich**. Explainable Predictive Monitoring of Temporal Measures of Business Processes. Tartu 2019, 151 p.
5. **Yauhen Yakimenka**. Failure Structures of Message-Passing Algorithms in Erasure Decoding and Compressed Sensing. Tartu 2019, 134 p.
6. **Irene Teinemaa**. Predictive and Prescriptive Monitoring of Business Process Outcomes. Tartu 2019, 196 p.
7. **Mohan Liyanage.** A Framework for Mobile Web of Things. Tartu 2019, 131 p.
8. **Toomas Krips.** Improving performance of secure real-number operations. Tartu 2019, 146 p.
9. **Vijayachitra Modhukur.** Profiling of DNA methylation patterns as biomarkers of human disease. Tartu 2019, 134 p.
10. **Elena Sügis.** Integration Methods for Heterogeneous Biological Data. Tartu 2019, 250 p.
11. **Tõnis Tasa.** Bioinformatics Approaches in Personalised Pharmacotherapy. Tartu 2019, 150 p.
12. **Sulev Reisberg.** Developing Computational Solutions for Personalized Medicine. Tartu 2019, 126 p.
13. **Huishi Yin.** Using a Kano-like Model to Facilitate Open Innovation in Requirements Engineering. Tartu 2019, 129 p.
14. **Faiz Ali Shah.** Extracting Information from App Reviews to Facilitate Software Development Activities. Tartu 2020, 149 p.
15. **Adriano Augusto**. Accurate and Efficient Discovery of Process Models from Event Logs. Tartu 2020, 194 p.
16. **Karim Baghery.** Reducing Trust and Improving Security in zk-SNARKs and Commitments. Tartu 2020, 245 p.
17. **Behzad Abdolmaleki.** On Succinct Non-Interactive Zero-Knowledge Protocols Under Weaker Trust Assumptions. Tartu 2020, 209 p.
18. **Janno Siim.** Non-Interactive Shuffle Arguments. Tartu 2020, 154 p.
19. **Ilya Kuzovkin.** Understanding Information Processing in Human Brain by Interpreting Machine Learning Models. Tartu 2020, 149 p.
20. **Orlenys López Pintado.** Collaborative Business Process Execution on the Blockchain: The Caterpillar System. Tartu 2020, 170 p.
21. **Ardi Tampuu.** Neural Networks for Analyzing Biological Data. Tartu 2020, 152 p.

22. **Madis Vasser.** Testing a Computational Theory of Brain Functioning with Virtual Reality. Tartu 2020, 106 p.
23. **Ljubov Jaanuska.** Haar Wavelet Method for Vibration Analysis of Beams and Parameter Quantification. Tartu 2021, 192 p.
24. **Arnis Parsovs.** Estonian Electronic Identity Card and its Security Challenges. Tartu 2021, 214 p.
25. **Kaido Lepik.** Inferring causality between transcriptome and complex traits. Tartu 2021, 224 p.
26. **Tauno Palts.** A Model for Assessing Computational Thinking Skills. Tartu 2021, 134 p.
27. **Liis Kolberg.** Developing and applying bioinformatics tools for gene expression data interpretation. Tartu 2021, 195 p.
28. **Dmytro Fishman.** Developing a data analysis pipeline for automated protein profiling in immunology. Tartu 2021, 155 p.
29. **Ivo Kubjas.** Algebraic Approaches to Problems Arising in Decentralized Systems. Tartu 2021, 120 p.
30. **Hina Anwar.** Towards Greener Software Engineering Using Software Analytics. Tartu 2021, 186 p.
31. **Veronika Plotnikova.** FIN-DM: A Data Mining Process for the Financial Services. Tartu 2021, 197 p.
32. **Manuel Camargo.** Automated Discovery of Business Process Simulation Models From Event Logs: A Hybrid Process Mining and Deep Learning Approach. Tartu 2021, 130 p.
33. **Volodymyr Leno.** Robotic Process Mining: Accelerating the Adoption of Robotic Process Automation. Tartu 2021, 119 p.